Classificação de Documentos Científicos Usando Modelos de Recuperação da Informação

Richard Mateus Hespanholo

Universidade Federal de Uberlândia
Faculdade de Computação
Bacharelado em Sistemas de Informação

Monte Carmelo - MG
2020

Dedico este trabalho primeiramente a Deus, por ser essencial em minha vida, autor de meu destino, meu guia, socorro presente na hora da angústia, ao meu pai Leandro Gabioli Hespanholo, minha mãe Claudia Mateus e ao meu irmão Rhyan Mateus Hespanholo. Dedico ao meu professor Carlos Cesar Mansur Tuma e aos meus colegas que me apoiaram na conclusão deste trabalho.

Agradecimentos

A Deus por ter me dado saúde e força para superar as dificuldades.

A esta universidade, seu corpo docente, direção e administração que deram esta oportunidade que vislumbro hoje.

Ao meu orientador Carlos Cesar Mansur Tuma, pelo suporte no concorrido tempo que lhe coube, pelas suas correções, incentivos e ideias.

Aos meus pais, pelo amor, incentivo e apoio incondicional.

E a todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

"Não confunda derrotas com fracasso nem vitórias com sucesso. Na vida de um campeão sempre haverá algumas derrotas, assim como na vida de um perdedor sempre haverá vitórias. A diferença é que, enquanto os campeões crescem nas derrotas, os perdedores se acomodam nas vitórias."

(Roberto Shinyashiki)

Resumo

Mecanismos de buscas como Google Scholar e Microsoft Academic, tidos como clássicos, apresentam uma deficiência ao classificar os resultados, utilizando-se de vários fatores externos aos conteúdos dos documentos, o que leva a uma classificação de resultados não interessante ao usuário. Este trabalho foi desenvolvido com o propósito de criar uma aplicação que classifique por relevância uma coleção de documentos de um repositório, em formato PDF ou TXT. O processo proposto extrai o conteúdo textual dos documentos, aplica várias técnicas de pré-processamento de Recuperação de Informação, modela na estrutura Bag of Words, aplica o modelo Vetorial com as métricas TF e IDF. A esta aplicação foi adicionado um dicionário de sinônimos a ser preenchido pelo usuário. No momento das buscas a aplicação expande a busca baseada no dicionário. Esta aplicação não considera os fatores externos ao conteúdo dos documentos e busca reduzir o tempo de pesquisa dos usuários por meio de uma classificação mais eficiente. Os resultados obtidos indicam que se alcançou o objetivo.

Palavras-chave: Recuperação da informação, Modelo de espaço vetorial, Mecanismos de busca, Expansão de consulta, Tempo de analise de documentos.

Lista de ilustrações

Lista de siglas

CVS Síndrome da Visão de Computador - *Computer Vision Syndrome*

DL Aprendizado Profunda - *Deep learning*

IA Inteligência Artificial - *Artificial Intelligence*

IDF Frequência Inversa do Documento - *Inverse document frequency*

ML Aprendizado de Máquina - *Machine Learning*

MSN Serviços de Mensagens da Microsoft - *Microsoft Service Network*

PDF Formato de Documento Portátil - *Portable Document Format*

PeD Pesquisa e Desenvolvimento - *Research and Development*

RI Recuperação da informação - *Information Retrieval*

TF Frequência de Termo - *Term Frequency*

TF-IDF Frequência do Termo-Inverso da Frequência nos Documentos - *Term Frequency-Inverse Document Frequency*

TXT .txt - *Arquivo de texto sem formatação*

VSM Modelo de espaço vetorial - *Vector Space Model*

Sumário

Capítulo 1

Introdução

No fim da década de 60, os primeiros sistemas de recuperação da informação foram desenvolvidos com base na ideia de organização de bibliotecas para acesso ao acervo de livros, artigos, periódicos e outros documentos (Sanderson; Croft, 2012).

A partir da década de 90 com a popularização da Internet, sistemas de indexação e buscas de páginas Web foram surgindo, como Google (About Google, 2019) e (Yahoo, 2019). Tais sistemas são utilizados em diversas áreas para diferentes fins, como buscas da Internet, buscas em registros de sistemas, acervos de bibliotecas, buscas em documentos em linguagem natural entre outros (Araújo Júnior, 2005).

Nos dias atuais, mecanismos de busca como Google indexam praticamente todo conteúdo da Web para facilitar a busca pelos usuários, usando robôs que vasculham as páginas para indexar todo seu conteúdo (Support Google, 2019). Neste contexto ele se torna uma das ferramentas mais usadas na pesquisa científica principalmente em universidades para encontrar artigos, publicações e etc., para embasamento de trabalhos da comunidade acadêmica e disseminação de conteúdo (Alison J. Head, 2007).

1.1 Motivação

A Recuperação da informação (RI) tem ênfase em desenvolver e estudar os processos de busca em um sistema computacional onde dado uma base de documentos, ser capaz de retornar um conjunto que atenda o usuário, avaliando assim, o desempenho dessas buscas. O conjunto resultado deve ser coerente, ordenado por uma classificação e relevante ao usuário. As informações obtidas são conhecidas como necessidade de informação do usuário (Yates; Neto, 2013).

Com a Internet e o acesso amplo a várias informações, em uma pesquisa científica acabam sendo selecionados diversos documentos para serem analisados sobre o tema em questão. Nem todos os documentos são relevantes e a busca dos relevantes implica leitura dos mesmos, aumentando o tempo de pesquisa e desenvolvimento.

Segundo o Google Trends (Google Trends, 2006) os usuários geralmente buscam com um número limitado de termos, algo em torno de 3 termos. Estas buscas resultam em listas de resultados de baixa qualidade, devido às poucas informações apresentadas. Este fato leva os mecanismos de busca a usarem técnicas como relevância de um site ou documento com base no número de acessos a ele. Isto acontece para fazer um refinamento da busca, já que podem existir milhares de documentos e sites que contenham este número limitado de termos de busca (Carpineto; Romano, 2012)

O Google considera documentos científicos como sendo semelhantes à páginas Web, assim seu formato de indexação e a recuperação destes documentos são semelhantes. Seu sistema de classificação é composto por vários algoritmos que buscam e analisam vários aspectos do usuário na hora de fazer uma busca. São utilizados vários métodos para ranquear um documento ou uma página Web, os rastreadores do Google processam o conteúdo do documento detectando sinais como palavras-chave e data de publicação do conteúdo em um índice (Google Webmaster Central Blog, 2011).

Na academia, são desenvolvidos um grande número de pesquisas em diversas áreas e de diversos tipos, desde robótica, Aprendizado de Máquina (ML), Aprendizado Profunda (DL), Inteligência Artificial (IA), algoritmos genéticos, pesquisas genéticas para plantas, vida marinha e mais outros diversos temas possíveis sendo pesquisados por alunos de graduação, pós-graduação, mestrado e doutorado. Estes pesquisadores em universidades e empresas buscam conhecimento, referências e direcionamento para realização de Pesquisa e Desenvolvimento (PeD) em livros, artigos, revistas e periódicos em sua maioria na Internet.

Na Internet temos diversos acervos disponíveis e mecanismos de pesquisa como Google, Google Acadêmico, Yahoo, Bing, dentre outros menos utilizados. Naturalmente um pesquisador que está desenvolvendo um trabalho, utiliza esses mecanismos para fazer suas pesquisas, adquirir conhecimentos e encontrar novas formas de abordagem e de pensamento sobre um tema.

Em mecanismos como Google, para cada pesquisa, um sistema de sinônimos é utilizado para expandir a consulta. Utilizado a cada termo relevante da consulta, como o termo "trocar" pode significar trocar uma lâmpada, ou ainda, trocar de roupa. De acordo com seu histórico de buscas, os sinônimos de alguma palavra podem ser diferentes, implicando em respostas diferentes.

Sua localização, palavras chave, histórico de buscas, sinônimos, idioma, data de publicação, número de acessos, relevância da página, número de citações (artigos) e se essas citações foram em publicações relevantes, são alguns dos fatores que as pesquisas feitas em ferramentas como Google levam em consideração (Google Webmaster Central Blog, 2015). Outras dezenas de fatores são citados por DEAN em sua publicação (DEAN, 2020).

Beel; Gipp realizaram uma pesquisa para determinar o impacto de alguns fatores

nas buscas no Google Acadêmico. Ele levou em consideração a contagem de citações no artigo, idade do artigo, ocorrência do termo no texto completo, frequência do termo no texto completo, ocorrência do termo no título do artigo e ocorrência do termo no nome do autor ou publicação e outros aspectos utilizado pelo estudo (Beel; Gipp, 2009).

Devido a esses fatores utilizados para ranquear, ou seja, classificar uma busca, artigos podem ser mais bem ranqueados, mas contendo um conteúdo menos relevante ao cientista, podendo ter uma distorção na classificação exibindo documentos que podem não ser interessantes ou até resultados tendenciosos.

Dessa forma, o número de variáveis e o número imenso de documentos presentes nas bases de pesquisas online podem afetar negativamente o ranqueamento de documentos científicos, artigos, publicações e periódicos. Essas características podem fazer com que os resultados não interessantes ou tendenciosos possam ter uma melhor pontuação na classificação, não satisfazendo o usuário. Como essa mistura de artigos tendenciosos ou não relevantes pode ser bem classificada, o número de artigos que precisam ser analisados para se encontrar por exemplo 10 documentos relevantes, pode aumentar, levando a utilização de mais tempo em leitura. Esse maior tempo na frente de equipamentos eletrônicos pode gerar problemas de saúde para o pesquisador, como Síndrome da Visão de Computador (CVS) que pode causar fadiga ocular e visão turva como conta Blehm et al..

1.2 Justificativa

De acordo com (Orduna-Malea et al., 2014), uma busca é feita em um acervo de milhões de artigos, certamente irá retornar um número considerável de resultados, mais de 2,8 milhões de resultados ao buscar “filosofia” no Google Acadêmico por exemplo. É humanamente impossível a leitura de tamanho número de artigos e documentos científicos.

Esses resultados normalmente são ranqueados por um conjunto interno de parâmetros pelo buscador, gerando um ranqueamento que tende a colocar os relevantes nos primeiros lugares. Tendo isso em vista, os primeiros resultados são interessantes, mesmo assim ainda há muitos artigos. Os 20 primeiros podem ser mais relevantes, porém os fatores discutidos anteriormente nos mecanismos de busca podem fazer com que alguns dos artigos de muita relevância se encontrem no fim da lista.

O pesquisador provavelmente deverá ler ou analisar todos os artigos científicos para descobrir que somente alguns deles são relevantes. Isso demonstra que perde-se um tempo considerável gasto na leitura de tais documentos científicos para selecionar os mais relevantes.

Com o menor tempo gasto para leitura de um conjunto de artigos para uma determinada busca, o pesquisador pode ler mais artigos relevantes em menos tempo, ampliando o número de artigos científicos analisados, otimizando o tempo e podendo melhorar a qualidade do trabalho.

Além da melhora da qualidade do trabalho, temos a otimização de tempo, pois podemos diminuir o período total em que o pesquisador analisa e seleciona artigos científicos. Dessa forma evitando problemas de saúde ocasionados por permanecer muitas horas sentado (Barros; Ângelo; Uchôa, 2011).

Problemas na coluna e dores musculares são muito comuns em pessoas que permanecem sentadas por muito tempo na mesma posição. Além da permanência em frente a uma tela, que em exposição prolongada pode ter efeitos negativos na qualidade do sono (BBC, 2015) e na visão do pesquisador (Blehm et al., 2005).

Com o objetivo de sanar tais deficiências propõe-se desenvolver uma aplicação que classifique por relevância uma coleção de documentos de um repositório, fazendo um refinamento desconsiderando os vícios encontrados.

De acordo com os fatores levantados, as pesquisas realizadas em mecanismos de buscas utilizam muitos aspectos para classificar os documentos trazendo resultados tendenciosos, sendo algumas dessas métricas *h-index* e *h-median*, que relacionam citações entre artigos que aumenta a influência destes nos resultados.

1.3 Objetivos

Para isso este trabalho foi desenvolvido com o propósito de criar uma aplicação que classifique por relevância uma coleção de documentos de um repositório, em formato PDF ou TXT. O processo proposto extrai o conteúdo textual dos documentos, aplica várias técnicas de pré-processamento de Recuperação de Informação, modela na estrutura *Bag-Of-Words*, aplica o modelo Vetorial com as métricas TF e IDF. A esta aplicação foi adicionado um dicionário de sinônimos a ser preenchido pelo usuário. No momento das buscas a aplicação expande a busca baseada no dicionário. Esta aplicação não considera os fatores externos ao conteúdo dos documentos e busca reduzir o tempo de pesquisa dos usuários por meio de uma classificação mais eficiente.

Com essa classificação a ordem de relevância dos artigos será aprimorada de acordo com o conteúdo, trazendo documentos mais relevantes com uma melhor classificação e aumentando a qualidade da busca e as chances de melhores resultados e diminuindo a necessidade de leitura de mais artigos.

1.4 Metodologia

Foi desenvolvido uma aplicação que tem a capacidade de carregar documentos, adicionar um dicionario de sinônimos e realizar uma classificação usando as ferramentas descritas na Seção 2.5, Escolhas do Trabalho. A aplicação foi distribuída nos sistemas operacionais Linux e Windows.

A avaliação da aplicação foi feita através de um formulário disponibilizado, onde os usuários avaliaram o desempenho da classificação. E feito uma analise desta avaliação para validar o desempenho da classificação.

1.5 Contribuições

A principal contribuição deste trabalho é criar uma solução capaz de melhorar a classificação de buscas nos mecanismos clássicos de busca, otimizando o tempo gasto com leituras de artigos e melhorando a qualidade do trabalho do pesquisador.

1.6 Organização do trabalho

No Capítulo 2 iremos abordar a Revisão bibliográfica para auxiliar na compreensão do trabalho. No Capítulo 3 é apresentada a implementação do mecanismo de busca e a utilização da aplicação. No Capítulo 4 é apresentado os resultados obtidos no estudo e desenvolvimento. Finalizando com o Capítulo 5 contendo as conclusões levantadas a partir dos resultados obtidos.

Capítulo 2

Revisão bibliográfica

Neste capítulo iremos abordar os conceitos de Recuperação da Informação, modelos, técnicas, ambiente de desenvolvimento e opções para o desenvolvimento deste trabalho.

2.1 O que é Recuperação da Informação

Recuperação da informação (RI) é uma das áreas da computação que lida com armazenamento e recuperação de documentos. O termo foi inicialmente criado por Calvin Mooers por volta dos anos de 1950 (Monteiro et al., 2017).

Tem como objetivo recuperar informação útil ao usuário a partir de uma coleção de documentos. Esses documentos geralmente possuem dados não estruturados, ou seja, não possuem uma tabulação ou classificação de seus termos ou campos bem definidos (Jahn, 2017). Em sua maioria, são documentos em linguagem natural por isso é passível de tolerância de pequenos erros e seus resultados não possuem uma resposta 100% válida.

O resultado de um sistema de RI é considerado relevante se ele atende as expectativas do usuário, sendo capaz de exibir uma classificação de documentos ordenados por relevância, ao qual o mais relevante deverá ser o documento que melhor atenda ao usuário.

RI é uma área bastante empírica, abrindo possibilidades há muitas formas diferentes de se abordar um problema. Boas ideias podem ser empregadas a esses sistemas fazendo-os ter um bom sucesso, como mecanismos de buscas como Google, Bing, etc. (Gomes; Cendón, 2015).

Na figura 1, vemos os campos da RI, a diferenciação dos campos entre "busca"e recuperação do conjunto de modelos ou algoritmos que foram evoluindo a partir de outros modelos. Para este trabalho foi abordado a recuperação Adhoc e seus modelos clássicos.

A partir da recuperação Adhoc, evoluiu dois conjuntos de modelos, os Clássicos e os Estruturados. Para os Clássicos, podemos dividir em três, os modelos do tipo Booleano, Espaço Vetorial e Probabilístico. Cada qual criando uma categoria de modelos própria com variações de algoritmos.

Redes de inferência e Redes de crença, foram uma evolução do modelo Probabilístico. Onde são chegados aos resultados através de cálculos probabilísticos. Semântica Latente, Redes Neurais e E. V. generalizado são evoluções do modelo Espaço Vetorial, que utiliza cálculos algébricos para chegar aos resultados. Já as técnicas Fuzzy e Booleano estendido são evoluções do modelo Booleano, que utiliza a teoria dos conjuntos para chegar aos resultados.

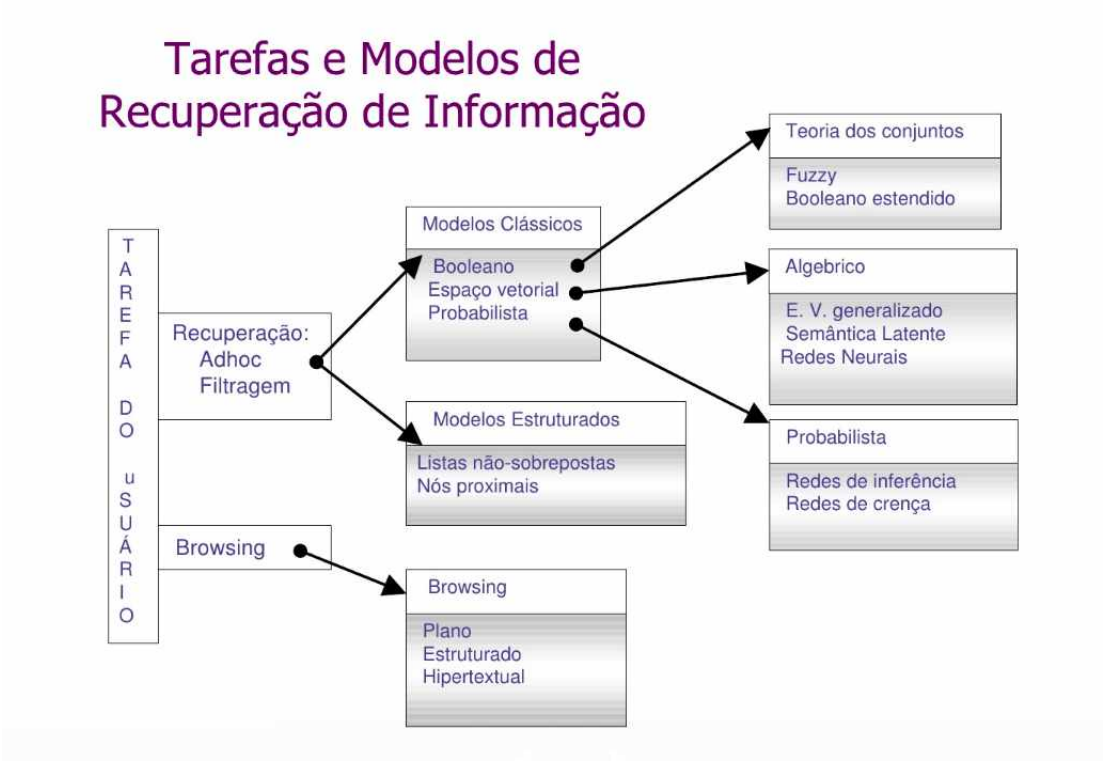

Figura 1 – Tarefas e Modelos de Recuperação de Informação. Autor: Nilton Heck

Fonte: https://pt.slideshare.net/niltonheck/aula02-recuperaodainformaomodelosdesistemasderecuperao

2.2 Conceito de Modelos de Recuperação da Informação

Existem várias abordagens nos modelos de RI que utilizam desde simples técnicas como modelo booleano até a modelos mais complexos como redes neurais ou a utilização de várias técnicas ao mesmo tempo.

Como apontado na Figura 2, tem-se uma representação simples das interações presentes em um mecanismo de busca. Duas entradas, uma de documentos e outra que necessita da intervenção do usuário, que seria a especificação da consulta. Na entrada de documentos, o mecanismo aplica um processo de indexação, que gera índices e uma representação dos documentos. Esses índices e a representação de documentos, mais a

consulta da necessidade do usuário, são entradas para o processo de recuperação, onde o algoritmo é aplicado gerando assim a lista de documentos recuperados.

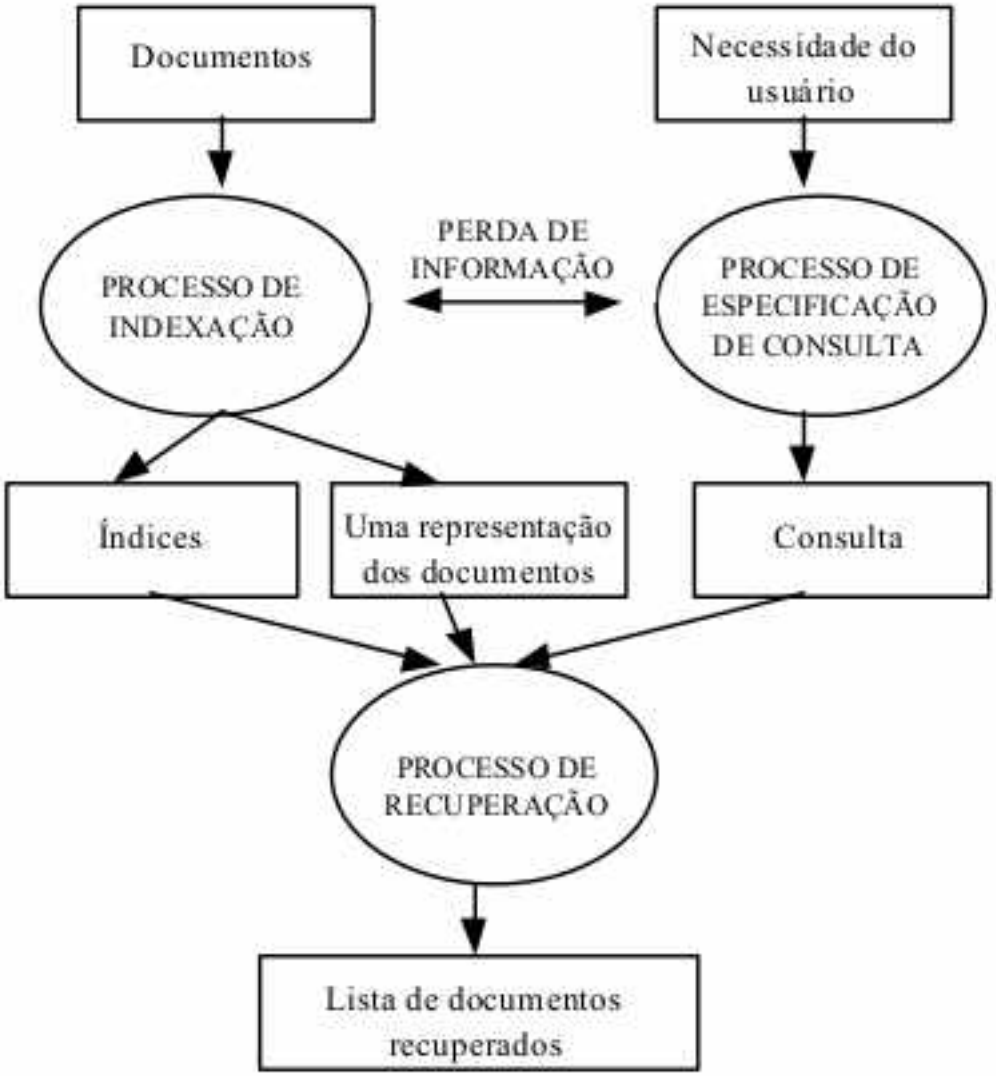

Figura 2 – Representação de um sistema de recuperação da informação. Autor: Olinda Nogueira Paes Cardoso

Fonte: http://www.dcc.ufla.br/infocomp/index.php/INFOCOMP/article/view/46

Como apontado por Baeza-Yates e pode ser visualizado pela Figura 2, existem alguns passos básicos que a maioria dos mecanismos de RI seguem:

1. passo: seleção dos documentos

   A base de dados geralmente são documentos em linguagem natural e não estruturados, podendo ser uma coleção de artigos em PDF ou apenas parágrafos de um artigo ou página. A seleção de documentos pode ser manual, onde o usuário seleciona um conjunto de documentos salvos no computador ou em outro local, para que

o mecanismo possa acessá-los ou de forma automática como acontece em sistemas de busca web que utilizam de *web crawlers* (MORAES, 2018) ou *robots* (GOOGLE, 2018) em um conjunto de páginas web para busca do conteúdo de cada página.

2. passo: preparação dos documentos

   Algumas regras podem ser aplicadas aos documentos como remoção de caracteres especiais, acentuação, pontuação, números e a retirada de *stopwords* (SAIF et al., 2014), que são palavras que não trazem um significado ao documento como, a, ou, para, um, uma (Figura 3). Também há a utilização de *stemmer* (Figura 4), que é transformar as palavras em seu radical, usado para eliminar variações de mesma palavra como favorito, favorita e favoritos, trazendo uma melhoria para os sistemas de RI (Flores; Moreira, 2016)

Sample text with Stop Words	Without Stop Words
GeeksforGeeks – A Computer Science Portal for Geeks	GeeksforGeeks , Computer Science, Portal ,Geeks
Can listening be exhausting?	Listening, Exhausting
I like reading, so I read	Like, Reading, read

Figura 3 – Exemplo de remoção de stopwords. Autor: Arup Jyoti Dutta

Fonte: https://www.geeksforgeeks.org/removing-stop-words-nltk-python/

Form	Suffix	Stem
studies	-es	studi
studying	-ing	study
niñas	-as	niñ
niñez	-ez	niñ

Figura 4 – Exemplo de stemming, transformação das palavras em seu radical. Autor: Vinicius dos Santos

Fonte: https://www.computersciencemaster.com.br/2018/12/aula-03-stemming.html

3. passo: indexar e representar a base de dados

   Uma vez com os documentos selecionados, é necessária uma estrutura eficiente para armazenar e buscar o conteúdo do documento. Pode-se representar um documento através de um *Bag-of-Words*, Figura 5, ou *Index* invertido, Figura 6, os quais estão explicados no tópico 2.2.1, Representação dos Documentos.

4. passo: definir modelo para elaborar consultas

Consiste em utilizar um modelo para ranquear os documentos de acordo com a consulta dada pelo usuário. Existem diversos métodos para ranqueamento de documentos como os modelos clássicos, estruturados, teoria dos conjuntos, algébricos e probabilístico.

Com esses passos pode-se montar um mecanismo de recuperação da informação capaz de ranquear documentos em linguagem natural de uma coleção de acordo com termos de busca do usuário.

2.2.1 Representação dos documentos

Para um sistema de recuperação, os documentos devem ser representados de forma que o computador possa interpretar e diferenciar cada documento dentro da coleção. Esta representação pode ser feita de varias formas. As principais representações de documentos são: *Bag-of-Words* e *Index* invertido:

- *Bag-of-Words*

  Nessa representação é feito uma separação de todos os termos do documento e armazenando em uma estrutura de lista, assim tem-se um *Bag* ou seja, um saco de palavras que representa o documento onde a ordem dos termos não é levado em consideração e o documento é representado por um vetor que contem todos os termos contidos na base de documentos, criando-se um vocabulário da coleção, onde atribui-se a frequência do termo naquele documento (BROWNLEE, 2017) e (SINGH, 2019).

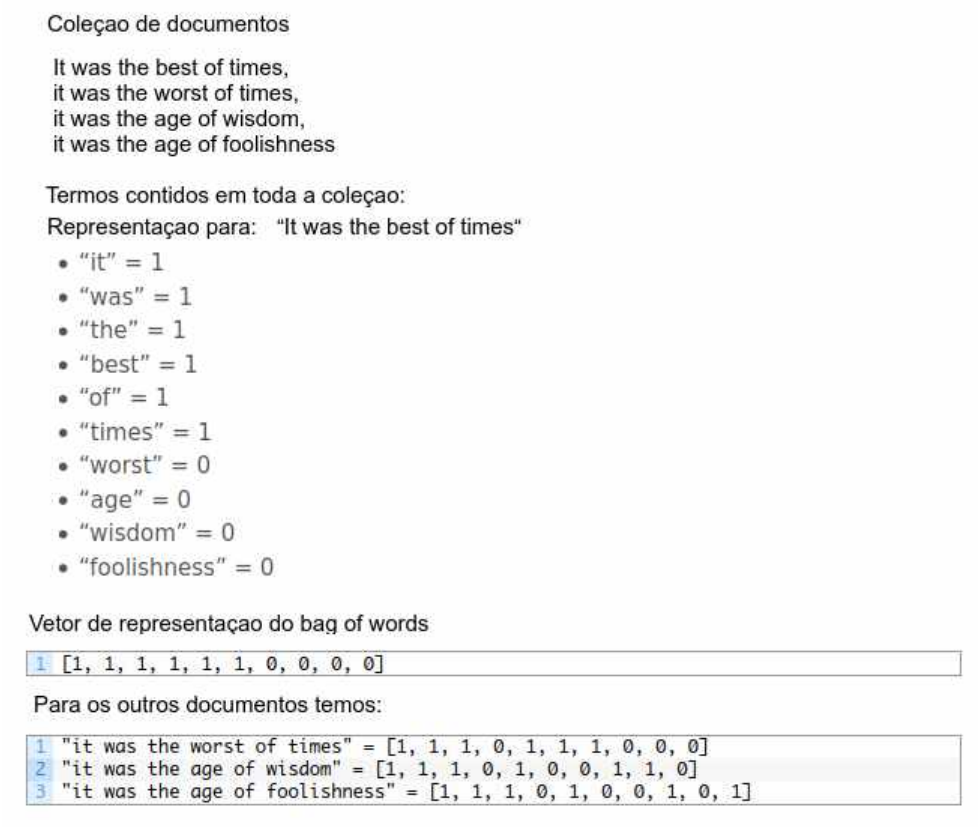

Figura 5 – Exemplo de representação dos documentos utilizando bag of words. Autor: Jason Brownlee

Fonte: https://machinelearningmastery.com/gentle-introduction-bag-words-model/

❑ *Index* invertido

Index invertido é uma representação simplificada de um conjunto de documentos, onde para cada termo presente na base de documentos, há uma lista informando em quais documentos e quantas vezes o termo esta contido no documento (Cambridge University Press, 2009). Assim temos uma busca mais rápida e eficiente na recuperação dos termos do documento, pois o acesso é direto ao termo, sem haver a necessidade de percorrer todos os termos até encontra-lo. Porém existe um custo, é mais trabalhoso a inserção e atualização dos documentos (XIA et al., 2013).

As principais diferenças entre *Bag-of-Words* e *Index* invertido, é que no *Bag-of-Words* os documentos são representados por uma lista com todos os termos da coleção e cada documento pode ser salvo em uma outra lista, tendo uma representação no formato de matriz. Já no *Index* invertido, temos que para cada termo da coleção é atribuído uma lista de quais documentos e quantas vezes aquele termo aparece no documento.

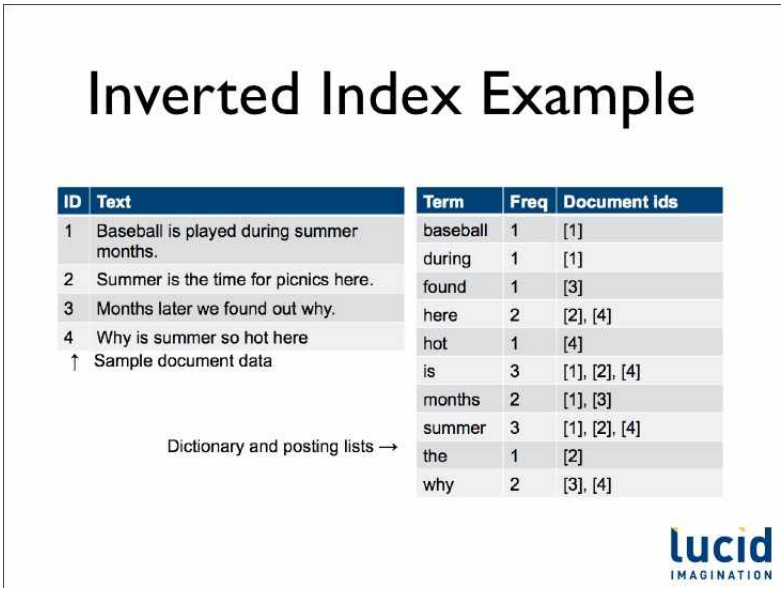

Figura 6 – Representação do index invertido. Autor: Erik Hatcher

Fonte: https://www.slideshare.net/erikhatcher/introduction-to-solr-9213241

2.2.2 Ponderação de termos

Usa-se ponderação de termos para medir o quão relevante é aquele termo para descrever um documento. Se um termo é muito relevante em toda a base de documentos, então ele não é tão relevante, pois não consegue descrever bem um documento.

Mas, se esse termo, é muito frequente em um documento e não na base toda, logo ele se torna relevante. Podendo descrever melhor aquele documento. Assim pode-se identificar o quão relevante é um termo dependendo do número de vezes que esse termo aparece em um documento e em toda a sua base de documentos (Cornell University Library, 1987).

Pode-se aplicar algumas medidas estatísticas para determinar o quão relevante é um termo, temos:

1. Frequência do Termo TF

   TF é a frequência com que um termo aparece em um documento. Ele leva em consideração que quanto mais um termo é frequente em um documento mais ele tem a capacidade de descrever este documento (ANALYTICS, 2019b).

   Sua implementação é simples, basta somar a quantidade de vezes que o termo aparece no documento.

2. Frequência Inversa do Documento IDF

   IDF é a frequência inversa do documento, visa medir o quão importante um termo é segundo sua raridade na base. Ou seja, quanto mais raro é um termo na base de documentos mais importante este termo é (ANALYTICS, 2019a).

$$w_{ij} = \begin{cases} 1 + \log f_{ij} & , \text{se } f_{ij} \geq 1 \\ 0 & , \text{caso contrário} \end{cases}$$

Figura 7 – Formula usada para calcular a frequência do termo. Onde 'W i,j' e o peso do termo 'i' no documento 'j'. e 'F i,j' é a frequencia do termo 'i' no documento 'j'. Autor: Wendel Melo

Fonte: http://www.facom.ufu.br/~wendelmelo/ori201902/4_ponderacao_de_termos.pdf

$$idf(k_i) = \log\left(\frac{N}{n_i}\right)$$

Figura 8 – Formula usada para calcular a frequência inversa do documento. Onde 'K i' é um termo dentro da base, 'N' é o total de documentos da coleção e 'N i' é o número de documentos que contem o termo 'K i'. Autor: Wendel Melo

Fonte: http://www.facom.ufu.br/~wendelmelo/ori201902/4_ponderacao_de_termos.pdf

3. Frequência do Termo - Frequência Inversa do Documento TF-IDF

   TF-IDF é a utilização do TF para determinar o quão relevante é aquele termo para o documento sendo mediado pelo IDF para determinar se aquele termo é relevante dentro de toda a base de documentos. Assim conseguimos medir o quão um termo é relevante para descrever um documento de acordo com sua frequência no documento e em toda a base de documentos (RAMOS et al., 2003).

$$w_{ij} = \begin{cases} tf(k_i,d_j) \times idf(k_i) & , \text{se } f_{ij} \geq 1 \\ 0 & , \text{caso contrário} \end{cases}$$

Figura 9 – Formula usada para calcular o TF-IDF. Onde para cada termo 'K i' multiplica-se o valor de TF e IDF retornando um peso 'W i,j' para cada termo da coleção. Autor: Wendel Melo

Fonte: http://www.facom.ufu.br/~wendelmelo/ori201902/4_ponderacao_de_termos.pdf

2.3 Modelos clássicos

Existem diversas abordagens para descobrir a similaridade entre textos, como os Modelos Estruturados e Clássico. O Modelo Estruturado pode se dividir em Teoria dos Conjuntos, Algébrico e Probabilista. Dentro do Modelo Clássico temos a abordagem

Booleana, Vetorial e Probabilista. Na sequencia, aborda-se sobre as divisões do Modelo Clássico.

2.3.1 Modelo booleano

O modelo booleano é baseado na teoria de conjuntos sendo capaz de realizar operações logicas do tipo *AND*, *OR* e *NOT*. Sua busca se da por satisfazer algumas dessas operações, retornando um conjunto de documentos. Por não atribuir um peso aos documentos, ou seja, uma classificação de relevância, todos os resultados devem ser considerados como uma solução que satisfaz a busca do usuário. Esta característica é uma das principais desvantagens do Modelo Booleano, pois os documentos só podem ser classificados como relevantes ou não relevantes (Lashkari; Mahdavi; Ghomi, 2009).

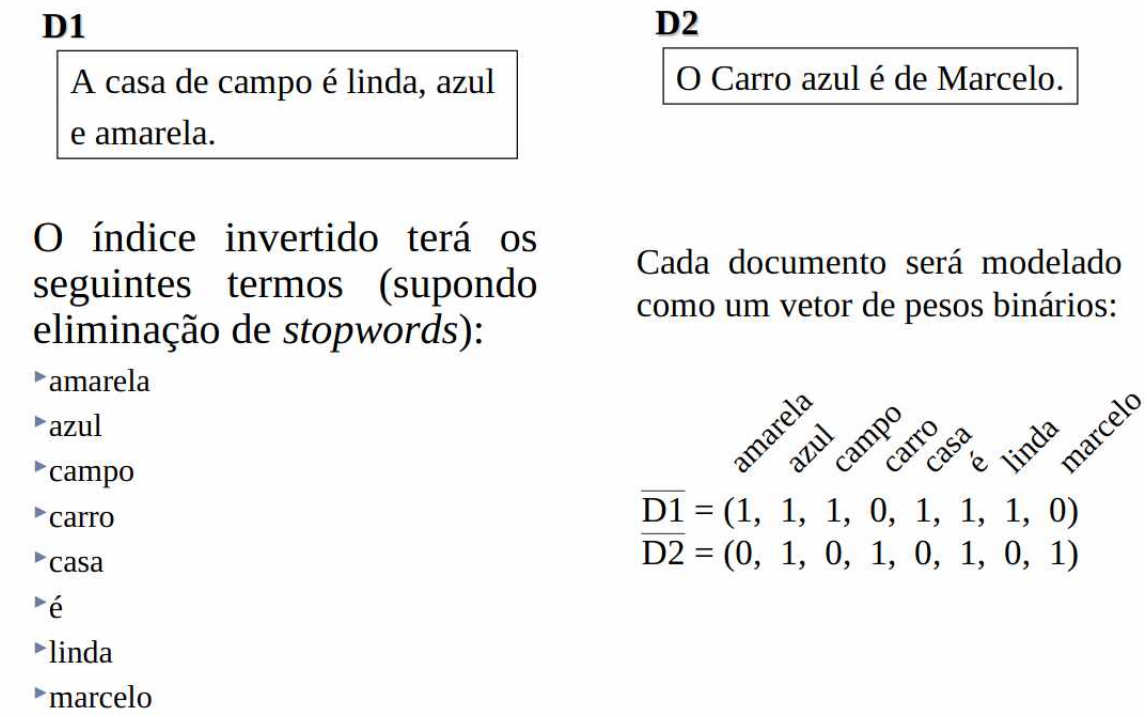

Figura 10 – Exemplo de modelo booleano. Autor: Wendel Melo

Fonte: http://www.facom.ufu.br/w̃endelmelo/ori201902/3_modelo_booleano.pdf

2.3.2 Modelo vetorial

Este modelo foi inicialmente proposto por *Salton* (Baeza-Yates, 2010). Ele representa documentos como um vetor, onde cada posição é uma palavra contida no documento. Todos os termos contidos em uma coleção de documentos, são utilizados para representar cada documento. Se o termo não se encontra em um documento, este termo tem o peso 0.

Este modelo propõem o casamento parcial entre busca e documentos por não atribuir valores binários a cada termo, ou seja, 1 se existe aquele termo no documento e 0 caso contrário. É utilizado uma ponderação de termos como TF-IDF, que atribui um valor entre 0 e 1 que representa o quão importante é aquele termo para representar um documento (WONG; RAGHAVAN, 1984).

2.3.3 Modelo probabilístico

O Modelo Probabilístico inicialmente desenvolvido por Roberstson e Sparck Jones parte do principio de que sempre existe um conjunto ideal que satisfaz a consulta do usuário. Sendo o seu objetivo, aproximar-se o máximo deste conjunto ideal (GARCIA, 2009).

Seu funcionamento básico se da por uma consulta inicial que resulta em um conjunto de documentos classificados. O usuário então indica os documentos relevantes. Assim, se repete o processo de classificação utilizando agora as informações obtidas pelas indicações do usuário para dar mais ênfase em documentos semelhantes aos selecionados. Dessa forma este modelo é caracterizado pelo seu funcionamento interativo com o usuário mas mesmo assim, pode-se utilizar interações automáticas que não necessitem da interação do usuário.

Uma de suas vantagens é essa interação com o usuário, fazendo com que os resultados reflitam as características de busca do usuário tornando o resultado mais próximo daquilo que o usuário espera. E os documentos são ordenados por ordem decrescente de relevância para o usuário.

Suas desvantagens são que na primeira interação é necessário gerar valores aleatórios para o cálculo das probabilidades. Isto também impacta negativamente caso sejam utilizados interações automáticas, pois os resultados vão depender de uma boa interação inicial para ter resultados melhores. Além disso, não é utilizado ponderação de termos neste modelo, fazendo com que ele desconsidere fatores como a relevância de um termo para o documento e dentro de toda a base de documentos (CROFT, 1981).

2.3.4 Expansão de consulta

A expansão de consulta é um processo que visa melhorar o desempenho de um sistemas de Recuperação da Informação através de técnicas aplicadas a busca do usuário. A utilização de expansão de consultas aumenta o número de termos de uma busca com o objetivo de corresponder a mais documentos da coleção, pois cada documento pode conter muitos termos enquanto a busca corresponde a uma pequena fração dos termos que podem estar presentes no documento (TUNKELANG, 2017).

Expansão de consulta por meio de sinônimos é uma técnica eficiente, que necessita da criação de um dicionário de sinônimos previamente inserida pelo usuário, podendo ser

especifico a uma determinada coleção ou um dicionário geral. A ideia é que para cada termo da busca e verificado no dicionário se existem sinônimos, caso existam então os termos são adicionados a consulta. Assim, pode-se aumentar o número de correspondências com os termos dos documentos da coleção (AFUAN AHMAD ASHARI, 2019).

Para expansão de consultas usa-se o algoritmo *Rocchio*. Este método é baseado em *feedback* de relevância, inicialmente desenvolvido por SALTON. Ele é baseado na ideia de que o usuário tem uma ideia de quais documentos podem ser relevantes ou não relevantes. Assim a pesquisa do usuário pode ser revisada utilizando partes dos documentos relevantes de forma a melhorar a recuperação do mecanismo de busca (MANNING; SCHüTZE, 2008).

2.4 Trabalhos Relacionados

No trabalho de SARKAR sobre Modelo de espaço vetorial (VSM), foi proposto utilizar uma adaptação do Modelo de Espaço Vetorial para encontrar uma relação entre doenças e seus genes causadores, onde muitos dos genes podem estar relacionados a várias doenças. Foram utilizados trés bases de conhecimentos para realização dos experimentos, *Online Mendelian Inheritance in Man*, *GenBank* e *Medline*. O estudo foi conduzido com alvo na doença de Alzheimer e síndrome de Prader-Willi.

Para o cenário utilizando a doença de Alzheimer, foram utilizados cinco genes como conjunto de consulta e seu resultado foi correto, mostrando como mais relevante a própria doença e em segundo lugar a doença renal policística que tem um papel associado a doença de Alzheimer. Um segundo cenário para a síndrome de *Prader-Willi* foi obtido como mais relevante a síndrome de *Angelman* que já e conhecida por atingir a mesma região cromossômica. Pelos resultados expostos, pode-se observar que está abordagem tem resultados promissores.

Baseado no Modelo de Espaço Vetorial, a abordagem utilizada por Castells; Fernandez; Vallet consiste em uma adaptação do modelo e uso de web semântica e explorando melhor as ontologias de um domínio especifico. Uma evolução do Modelo de Espaço Vetorial onde substitui índices baseados em palavras chave por uma base de conhecimento baseada em ontologia. Esta abordagem foi testada usando um corpus com 145.316 documentos e uma base de conhecimento desenvolvida pela *Ontotext Lab*.

Foram adotadas 20 perguntas testadas manualmente onde tem-se uma mistura de resultados que foram muito bem em perguntas onde se o uso da base de conhecimento melhora os resultados, porém em alguns casos não tem um bom resultado por não haver termos para representar alguns conhecimentos de tópicos que foram mencionados na consulta. No geral a utilização desta abordagem para recuperação de documentos científicos ou não se mostrou uma melhora em cima do Modelo de Espaço Vetorial. Porém a necessidade de uma base de conhecimento prévia pode diminuir o alcance de vários

conteúdos.

Outro trabalho usando o Modelo de Espaço Vetorial foi proposto por SINGH; DWIVEDI. Foi proposto uma abordagem a fim de melhorar o desempenho do modelo, levando em consideração o tamanho do documento menos o número de *stopwords* presentes no documento. Para avaliar essa abordagem, foi avaliado o desempenho de três mecanismos de busca, Google, Yahoo e MSN. Usando três métodos para a avaliação, o modelo proposto, VSM e uma avalização feita manualmente. Foram utilizados cinquenta perguntas e os dez primeiros documentos retornados por cada uma das perguntas foi aplicado os métodos para avaliar a similaridade. O método proposto obteve um bom desempenho similar ao encontrado na avaliação manual e o VSM clássico obteve um desempenho mais baixo mas sendo interessante. Ainda é necessário um maior amadurecimento do modelo proposto em comparação com o modelo VSM clássico, mostrando assim que o modelo em seu estado original tem um bom desempenho na similaridades de documentos mesmo em bases diferentes.

Um trabalho feito por PURBASARI; ANGGRAENY; MAHARANI, utilizou VSM para classificar perguntas e respostas ao estilo chatbot. Foi usado um conjunto de dados em forma de perguntas e seus resultados. Ao todo foram 42 alunos participantes e um total de 100 pares de perguntas e respostas dentro de um domínio de conhecimento especifico. Foram feitos três cenários de testes, no primeiro cenário tinha-se questões com o mesmo significado que as questões no banco de dados, mas em uma redação diferente. O segundo cenário consiste em buscas que possuem copia salva como perguntas dentro do banco de dados. Foram utilizados 5 questões para cada cenário, sendo os 10 principais resultados e suas questões armazenados em um banco de dados. O terceiro cenário, consiste em uma busca nessa base de dados gerada a partir do cenário 1 e 2. Todos os testes obtiveram um bom número de respostas relevantes, sendo que o teste do cenário 3 obteve uma menor número de documentos relevantes retornados. Podendo ser justificado pelo fato de utilizar o resultado de outras buscas poderem ter adicionado uma quantidade de documentos tendenciosa.

Esses trabalhos demonstram a eficiência do VSM em buscas dentro de conjuntos de documentos dos mais variados assuntos. Podendo ser utilizado para sistemas de buscas e respostas, semelhante a mecanismos de busca clássicos. Como visto no trabalho de PURBASARI; ANGGRAENY; MAHARANI por exemplo, com buscas em uma base de dados previamente manipulada, por usar os resultados de outros cenários, podem afetar a performance de um mecanismo de recuperação de informações.

2.5 Escolhas do trabalho

Analisando os vários conceitos descritos neste capítulo, foram feitas as seguintes escolhas para este projeto:

- Para cálculo da similaridade entre consultas e documentos, foi escolhido o Modelo de Espaço Vetorial, pois permite o casamento parcial entre buscas e documentos e não possui a necessidade de valores aleatórios ou avaliação por parte do usuário para refinamento das buscas.

- Para a representação dos documentos, foi escolhido *Bag-of-Words*, pois a representação dos documentos em vetores, com a contagem da frequência de cada termo consegue trazer ganhos ao realizar a ponderação dos termos.

- Para a ponderação de termos foi escolhido o TF-IDF pois balanceia o peso de cada termo utilizando-se da frequência do termo TF para demonstrar o quão relevante é aquele termo para representar aquele documento e o IDF que demonstra o quão relevante é aquele termo levando em consideração toda a base de documentos.

- Para expansão de consultas, foi utilizado o dicionário de sinônimos, pois não necessita de um *feedback* do usuário a cada busca feita e pode ser personalizado pelo usuário para atender as várias coleções de documentos da mesma área de conhecimento.

Capítulo 3

Desenvolvimento

A aplicação fornece uma interface onde o usuário fornecerá ao sistema um conjunto de documentos em formato PDF ou TXT. Este conjunto deve ser providenciado por meio de uma busca em algum mecanismo clássico de busca em que o tema da pesquisa não faz interferência no desempenho do sistema. A aplicação se encarrega de processar os documentos, aplicar alguns filtros como remoção de *stopwords*, aplicação de *stemming*, colocar todos os caracteres em *lowcase*, retirar os caracteres especiais e a partir disso calcular a similaridade com a busca feita pelo usuário, onde os mesmo filtros são aplicados, apresentando assim uma nova classificação dos documentos em ordem decrescente de relevância. Nesse capitulo serão explicados todos os passos acima:

A aplicação não tem um número máximo de documentos por coleção ou no numero de coleções definido, o limite teórico é de acordo com a capacidade do computador do usuário. Cada coleção deve conter documentos em apenas um idioma.

3.1 Algoritmo

O modelo utilizado neste trabalho, VSM, foi implementado usando diferentes bibliotecas, como *Scikit-Learn* para cálculo do cosseno de similaridade e a transformação de uma coleção de documentos em uma matriz com a contagem das palavras. *Nltk* para *stopwords*, extração do radical das palavras e quebra de todo o texto em uma lista de palavras, *tokens*. E *unicodedata* para normalização do texto, como retirada de caracteres especiais e deixar todo o texto em minúsculo.

3.1.1 Extração do texto

Todo o texto de um documento PDF foi extraído utilizando a função *extract_text* do modulo *high_level* da biblioteca *pdfminer* (Figura 11). Esta biblioteca consegue recuperar apenas o conteúdo textual contido no PDF ignorando elementos como imagens por exemplo, Figura 11.

```
def __pdf_to_txt(self, path):
    text = extract_text(path)
    return text
```

Figura 11 – Extração do texto do PDF

Para documentos TXT, foi utilizado apenas a função de leitura de arquivo de texto padrão *read*, já que documentos no formato TXT não possuem formatação, assim eles não necessitam de uma biblioteca extra, Figura 12.

```
def __read_txt(self, path):
    with open(path) as txt_file:
        text = txt_file.read()
    return text
```

Figura 12 – Extração do texto do documentos TXT

Para cada documento carregado, é executado o processo de extração de seu conteúdo, como mencionado anteriormente. O conteúdo é adicionado a uma lista com todos os documentos da coleção e em uma lista secundária é adicionado os nomes dos arquivos do documento. Cada posição das duas listas representa o mesmo documento, conforme apresentado na Figura 13.

```
for f in files:
    print(f'-- Carregando documento {f[0]}')
    if f[0].endswith('.txt'):
        self.documents.append(self.__read_txt(f[0]))
    else:
        self.documents.append(self.__pdf_to_txt(f[0]))
    self.documents_index.append(f[1])
```

Figura 13 – Extração e criação da lista de documentos

3.1.2 pré-processamento dos documentos

No momento que é gerado a representação da coleção, cada documento precisa passar por um pré-processamento. Através da biblioteca *nltk*, é feito a *tokenização* do documento usando a função *word_tokenize*, por exemplo, cada palavra de uma frase é um *token* quando um texto é *tokenizado* em palavras. Cada termo da lista passa por um processo de normalização, sendo usada a função *lower()* para transformar cada caractere em minúsculo e remoção de caracteres especiais usando a função normalize presente na biblioteca *unicodedata*, e é aplicada a redução da palavra ao seu radical, *stemming*, onde possui duas funções específicas para cada idioma suportado, *PortugueseStemmer* para português e *EnglishStemmer* para inglês. Além disso é removido as *stopwords* da lista de palavras através de um laço de repetição, Figura 14.

```
def pre_processing(doc):
    palavras = nltk.word_tokenize(
        normalize('NFKD', doc.lower()).encode('ASCII', 'ignore').decode('ASCII')
    )

    if language == 'portuguese':
        return (PortugueseStemmer().stem(w) for w in palavras if w not in stopwords)

    return (EnglishStemmer().stem(w) for w in palavras if w not in stopwords)
```

Figura 14 – pré-processamento dos documentos

3.1.3 Representação dos documentos

Através da função *CountVectorizer* presente na biblioteca *sklearn*, Figura 15, uma coleção de documentos pode ser convertida a uma matriz com a contagem *tokens*, formando um *Bag-Of-Words*.

```
self.cv = CountVectorizer(analyzer=pre_processing)

print(f'-- Criando modelo de processamento dos documentos...')
self.cv.fit_transform(self.documents)

print(f'-- Gerando vocabulario...')
self.vocabulary = self.cv.get_feature_names()

print(f'-- Gerando matriz de elementos processados dos documentos...')
self.matrix_corpus = self.cv.transform(self.documents).toarray()
```

Figura 15 – Instancia do processador de documentos, criação do vocabulário e matriz de representação dos documentos.

Todos os documentos da coleção já foram submetidos ao pré-processamento. Gerando uma representação da coleção e uma lista de todo o vocabulário da coleção que será usada no cálculo do TF-IDF, apresentado na Figura 16.

3.1.4 Ponderação de termos

Para realizar a ponderação dos termos de cada documento da coleção, primeiro deve ser calculado o valor do IDF de cada termo presente na coleção. Para esta finalidade, não foram utilizados bibliotecas para os cálculos sendo eles implementados na aplicação. O

```
# Coleção de documentos
documentos = [
    'This is the first document.',
    'This document is the second document.',
    'And this is the third one.',
    'Is this the first document?',
]

# Vocabulário gerado
['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third', 'this']

# Matriz de representação dos documentos
# com a contagem de termos
matriz = [[0 1 1 1 0 0 1 0 1]
          [0 2 0 1 0 1 1 0 1]
          [1 0 0 1 1 0 1 1 1]
          [0 1 1 1 0 0 1 0 1]]
```

Figura 16 – Lista de documentos, vocabulário gerado apos processamento e matriz de representação da coleção.

IDF e calculado para todos os termos presentes na coleção, afim de diminuir a relevância de termos muito frequentes.

O algoritmo executa um laço de repetição, e para todos os termos do vocabulário calculando o valor do IDF. Temos N, numero total de documentos e n que é o numero de documentos que contem aquele termo. Em seguida é calculado o logaritmo de N dividido por n. Caso tenha apenas um documento na coleção, o valor de n e incrementado em um, pois o logaritmo de *1/1* tem como resultado o valor zero, sendo assim nem um termo teria peso, impedindo a similaridade entre qualquer busca e este único documento. O valor resultante do logaritmo e adicionado a uma lista, sendo o valor com quatro casas decimais, Figura 17.

```
print(f'-- Calculando IDF dos documentos...')
self.idf = []
for i in range(len(self.vocabulary)):
    n = len([b for b in self.matrix_corpus if b[i] != 0])
    N = len(self.matrix_corpus)
    if N == 1:
        N += 1
    v_idf = math.log10(N/n)
    self.idf.append(round(v_idf, 4))
```

Figura 17 – cálculo do IDF

Logo em seguida é calculado o TF-IDF utilizando os valores calculados anteriormente no IDF. Este cálculo não foi utilizado bibliotecas, sendo implementado na aplicação. Um laço de repetição percorre todas as linhas da matriz, cada linha representa um documento, em seguida outro laço de repetição percorre os elementos da linha, as colunas da matriz, que representa os termos presentes no documento. Então é recuperado quantas vezes aquele termo aparece naquele documento, este é o TF. Em seguida é calculado o valor de TF, sendo 1 (um) mais o logaritmo do valor recuperado anteriormente. O resultado do TF e multiplicado pelo IDF daquele termo, o resultado então e adicionado a uma lista.

Ao final do cálculo do TF-IDF de todos os termos do documento, é gerando um *Bag-Of-Words* onde cada valor é a ponderação do termo calculada através do TF-IDF, Figura 18.

```
print(f'-- Calculando TF_IDF dos documentos...')
self.tf_idf = []
for corpus in self.matrix_corpus:
    aux = []
    for i in range(len(corpus)):
        f = corpus[i]
        if f == 0:
            tf = 0
        else:
            tf = 1 + math.log10(f)
        aux.append(round(tf * self.idf[i], 4))
    self.tf_idf.append(aux)
```

Figura 18 – Cálculo do TF-IDF

3.1.5 Processamento da busca

Ao realizar uma busca, é adicionado o dicionário de sinônimos a consulta, fazendo a sua expansão com os termos previamente cadastrados pelo usuário. Se os termos da consulta estão presentes no dicionário, então é adicionado esses novos termos. Em seguida é aplicado o mesmo pré-processamento utilizado na coleção e a mesma função *Count-Vectorizer* para gerar um *Bag-Of-Words* que representa o documento da consulta Figura 19.

```
print(f'-- Gerando matriz de elementos processados da busca...')
self.matrix_query = self.cv.transform([query]).toarray()
```

Figura 19 – Criação da matriz que representa os termos da busca.

Em seguida é calculado o TF-IDF da consulta utilizando o IDF calculado na ponderação dos termos da coleção. O cálculo do TF-IDF é realizado no *Bag-Of-Words* da consulta, conforme foi demonstrado na Figura 18.

3.1.6 Cosseno de similaridade

Apos os cálculos de ponderação dos termos da consulta e da coleção, é utilizado a biblioteca *sklearn* para calcular o cosseno de similaridade através da função *cosine_similarity*.

```
print(f'-- Executando o cosseno de similaridade...')
self.similarity = cosine_similarity(self.tf_idf_query, self.tf_idf)[0]
```

Figura 20 – cálculo do cosseno de similaridade

Os dois *Bag-Of-Words* gerados anteriormente são usados para os cálculos, gerando assim uma lista com o valor da similaridade entre cada documento da coleção com a busca, Figura 20. Cada valor desta lista é a similaridade que aquele documento tem com a consulta. Ao final é ordenado de forma decrescente os resultados e apresentados ao usuário, conforme apresentado na Figura 21.

```
print(f'-- Gerando resultados...\n')
response = []
for i in range(len(self.similarity)):
    response.append(
        (
            round(self.similarity[i] * 100, 2),
            self.documents[i],
            self.documents_index[i]
        )
    )

return response
```

Figura 21 – Ordenação dos resultados por ordem decrescente.

3.2 Utilização da aplicação

A aplicação foi desenvolvida tendo como interface de interação com o usuário por meio de entradas no teclado de acordo com as opções exibidas em tela. Ao iniciar a aplicação, o usuário tem a disposição um menu principal com quatro opções como mostrado na Figura 22.

Cada opção tem o seguinte objetivo:

- Fazer busca: o usuário seleciona uma coleção de documentos previamente cadastrada, seleciona se quer utilizar um dicionário de sinônimos, insere a busca e visualiza os resultados.

- Adicionar coleções: o usuário adiciona novas coleções, conjunto de documentos de um determinado tema ou pesquisa. Esses documentos podem ser no formato PDF e/ou TXT, no idioma inglês ou português. Apenas deve ser selecionado o local no seu computador onde esta os arquivos, o nome da coleção, o tipo de arquivo e o idioma.

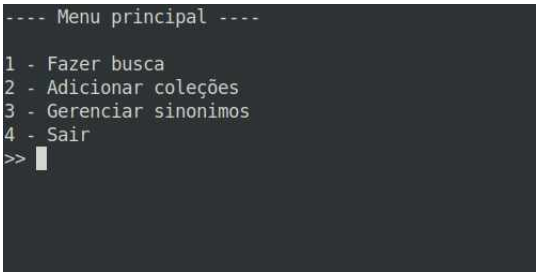

Figura 22 – Menu inicial

- Gerenciar sinônimos: o usuário pode gerenciar sinônimos, adicionar, remover ou visualizar.

- Sair: para sair da aplicação

Ao selecionar a opção 1 do menu principal, Fazer busca, ira listar as coleções salvas, Figura 23. O usuário deve selecionar qual a coleção desejada.

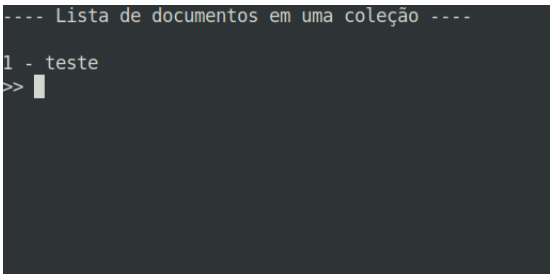

Figura 23 – Lista de coleções

Apos ele deve selecionar se usara dicionário de sinônimos para expandir a busca, Figura 24.

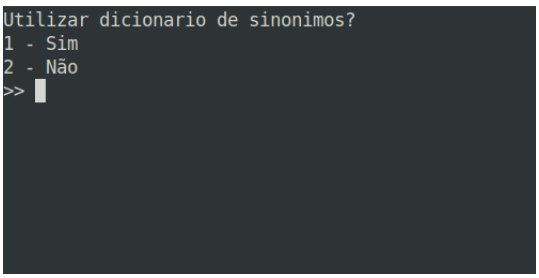

Figura 24 – Seleção de sinônimos

O usuário deverá digitar sua busca e aguardar o resultado que será apresentado, Figura 25.

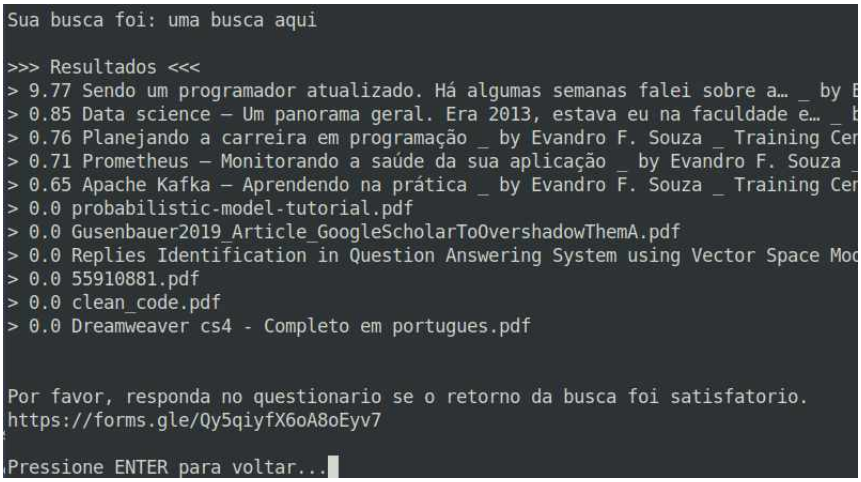

Figura 25 – Resultados da busca

Quando selecionado a opção 2 do menu principal, Adicionar coleções, o usuário deve selecionar a pasta onde contem os documentos utilizados para essa busca, Figura 26.

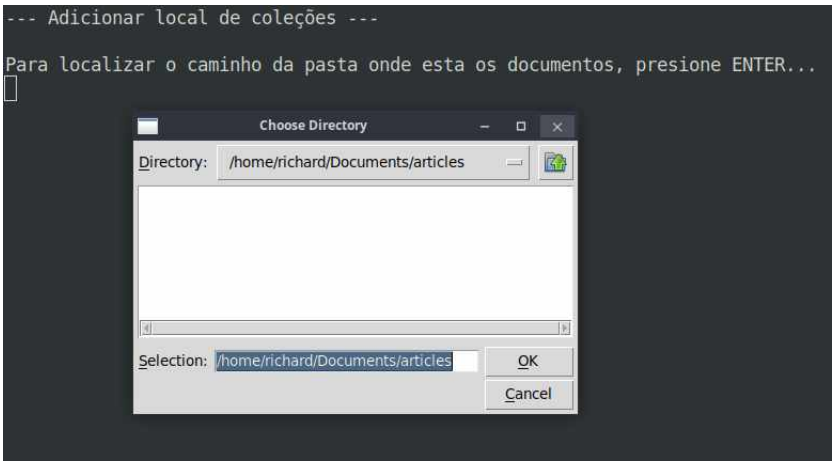

Figura 26 – Seleção de pasta dos documentos

Apos deve selecionar qual tipo de arquivo, PDF, TXT ou ambos, Figura 27.

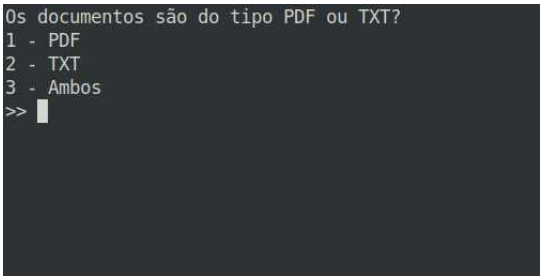

Figura 27 – Seleção do tipo dos documentos

Será listado todos os documentos reconhecidos e ira pedir o nome da coleção, Figura 28.

```
Documento: Dreamweaver cs4 - Completo em portugues.pdf

Documento: Apache Kafka – Aprendendo na prática _ by Evandro F. Souza _ Training Center _ Medium.pdf

Documento: Gusenbauer2019_Article_GoogleScholarToOvershadowThemA.pdf

Documento: Prometheus – Monitorando a saúde da sua aplicação _ by Evandro F. Souza _ Tech@Grupo ZAP _ Medium.pdf

Documento: 55910881.pdf

Documento: clean_code.pdf

Documento: probabilistic-model-tutorial.pdf


Insira um nome para esta coleção:
colecao 1
```

Figura 28 – Entrar com nome da coleção

Logo em seguida deve ser selecionado o idioma dos documentos, Português ou Inglês, Figura 29.

```
Qual o idioma dos documentos desta coleção?
1 - Português
2 - Inglês
>>
```

Figura 29 – Selecionar idioma dos documentos

Na opção 3 do menu principal, Gerenciar sinônimos, mostrará ao usuário um novo menu, Figura 30.

```
---- Dicionario de sinonimos ----

1 - Listar sinonimos
2 - Adicionar sinonimos
3 - Remover sinonimo
4 - Voltar
>>
```

Figura 30 – Menu de gerenciamento de sinônimos

A opção de Listar sinônimos, que lista todos os sinônimos cadastrados, Figura 31.

```
--- Dicionario de sinonimos ---

Palavra: limpo
Sinonimos: ['claro', 'brilhando']
--------//--------

Pressione ENTER para voltar...
```

Figura 31 – Lista de sinônimos cadastrados

Em Adicionar sinônimos, o usuário deve entrar com a palavra e em seguida será perguntado qual o sinônimo, Figura 32.

```
--- Adicionar ao dicionario de sinonimos ---

Qual palavra deseja adicionar?
limpo

--- Digite um sinonimo por vez e aperte ENTER ---

--- Ao final digite "sair" para não adicionar novos sinonimos ---

Qual sinonimo deseja adicionar?
brilhando

Sinonimo "brilhando"" foi adicionado.

Qual sinonimo deseja adicionar?
claro

Sinonimo "claro"" foi adicionado.

Qual sinonimo deseja adicionar?
sair
```

Figura 32 – Adiciona sinônimos

Remover sinônimos, será listado as palavras e o usuário pode digitar qual deve ser removida, Figura 33, ou voltar para ir ao menu principal.

```
--- Remover do dicionario de sinonimos ---

Palavra: limpo

Qual palavra deseja remover?
limpo

Pressione ENTER para voltar...
```

Figura 33 – Removendo sinônimos

Capítulo 4

Resultados

Neste capítulo são apresentados como foi obtido os resultados da avaliação da aplicação, assim com as questões apresentadas e uma visualização em gráficos dos resultados e sua análise.

4.1 Obtenção dos resultados

A aplicação foi distribuída junto com um tutorial explicando a instalação, utilização e link de um formulário Google disponibilizado para que os usuários avaliassem a aplicação. Abaixo, o tutorial disponibilizado e as perguntas do questionário:

4.1.1 Tutorial de instalação e utilização da aplicação

Objetivo da aplicação

A aplicação tem como objetivo realizar busca em uma coleção de documentos de forma a não ser influenciada por outros fatores como seu histórico de pesquisa, utilizado pelos mecanismos de busca clássico com Google e Bing.

Para isso foi utilizado o Modelo de Espaço Vetorial com expansão de consulta por sinônimos para realização das buscas em coleções de documentos.

Instalação

Linux
- Baixe o arquivo tcc-project-linux.tar.gz
- Extraia o arquivo tcc-project
- Abra o terminal na pasta em que foi extraído o arquivo

- Execute o comando ./tcc-project

Windows
- Baixe o arquivo tcc-project-windows.zip
- Extraia o arquivo tcc-project.exe
- De dois cliques no arquivo (O CMD irá abrir, pode levar cerca de 30 segundos até iniciar a aplicação.)

Modo de usar a aplicação

Ao iniciar a aplicação você terá a disposição 3 opções, Fazer busca, Adicionar coleções e Gerenciar sinônimos.

Inicialmente você deve adicionar uma coleção na opção 2 do menu. Mas para isto você deve estar de posse de documentos em PDF ou TXT. Estes arquivos devem ser obtidos da seguinte forma:

- Faça uma busca no Google.
- Analise os 10 primeiros resultados disponíveis para download
(De preferência tire um print da tela para saber a ordem e marque quais são os relevantes para você e em que ordem eles apareceram na busca do Google)
- Baixe os 10 primeiros resultados disponíveis em uma pasta apenas com esses documentos da busca
- Na aplicação vá na opção de Adicionar coleções e selecione a pasta em questão
- De um nome a coleção
- (Opcional) Para adicionar sinônimos que podem ajudar na busca vá na opção Gerenciar sinônimos
- Vá em Adicionar sinônimos e adicione uma palavra é os sinônimos que desejar
- Você também pode listar ou remover algum sinônimo da lista
- Apos isso vá em Fazer busca selecionar a opção com o nome da coleção
- (Opcional) Caso queira usar um dicionário de sinônimos, responda a opção correspondente (e necessário ter sinônimos adicionados).
- De preferência utilize a mesma busca que foi feita no Google
- NOTA: O tempo de busca varia de acordo com o número e tamanho dos documentos. Pode levar de 20 segundos até 5 minutos.
- Visualize o resultado em ordem decrescente de relevância
- Ao final verifique se os primeiros documentos retornados pela aplicação foram os mesmo que você considerou mais relevante da busca original no Google.

Respondendo o questionário

Ao final por favor responda o questionário disponível no link
https://forms.gle/Qy5qiyfX6oA8oEyv7

4.1.2 Questionário e resultados obtidos

O questionário disponibilizado possui 7 perguntas objetivas e 1 aberta para sugestão ou opinião.

Questionário Trabalho de Conclusão de Curso - Classificação de Documentos Científicos Usando Modelos de Recuperação da Informação

*Obrigatório

01) Você acha que um sistema que economizasse seu tempo para encontrar os documentos realmente relevantes é importante?*

a) Sim

b) Não

c) Talvez

Resultado:

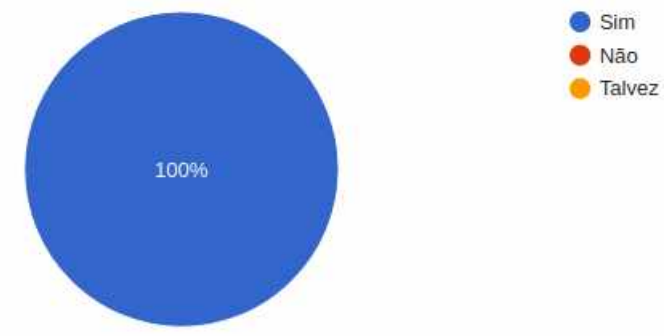

Figura 34 – Gráfico da questão 01 sobre a importância de um sistema que economize tempo nas buscas.

02) Por conta das revisões bibliográficas, você sofre com cansaço físico, visual e dores nas costas por passar muito tempo sentado em frente ao computador?*

a) Sim

b) Não

c) Talvez

Resultado:

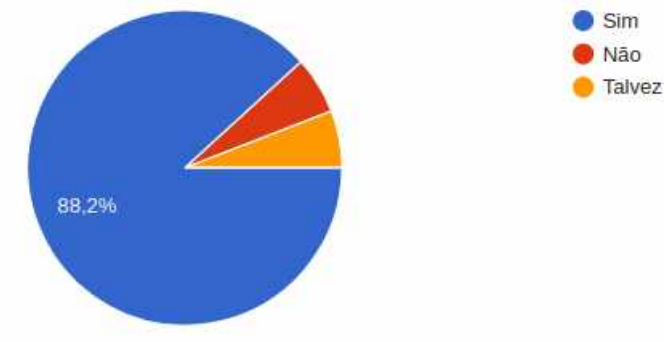

Figura 35 – Gráfico da questão 02 sobre se o usuário sofre de algum tipo de dor ou cansaço ao ficar muito tempo em frente a um computador.

03) Você considera que a classificação usando o aplicativo foi melhor que a fornecida pelo seu mecanismo de busca?*

a) Sim

b) Não

c) Talvez

Resultado:

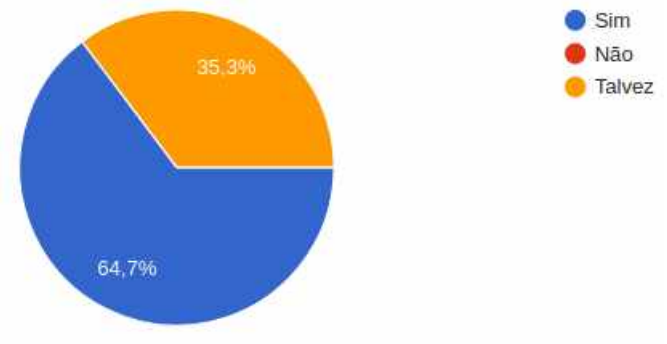

Figura 36 – Gráfico da questão 03 sobre a classificação da aplicação ter tido melhor desempenho que o mecanismo de busca.

04) Você considera que a classificação usando o aplicativo mais o uso de sinônimos foi melhor que a fornecida pelo seu mecanismo de busca?*

a) Sim

b) Não

c) Talvez

Resultado:

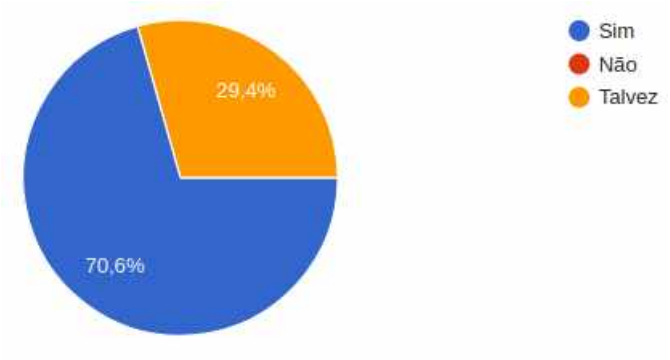

Figura 37 – Gráfico da questão 04 sobre o desempenho da classificação utilizando sinônimos.

05) Você considera que com o uso do aplicativo poderá reduzir o tempo gasto lendo os documentos na frente no computador?*

a) Sim

b) Não

c) Talvez

Resultado:

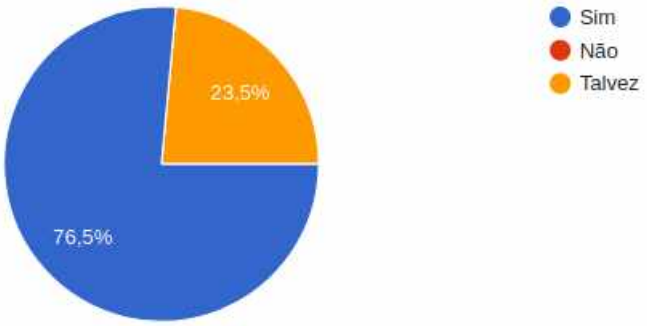

Figura 38 – Gráfico da questão 05 sobre se a aplicação pode reduzir o tempo na frente do computador.

06) Você considera o aplicativo de fácil instalação?*

a) Sim

b) Não

c) Talvez

Resultado:

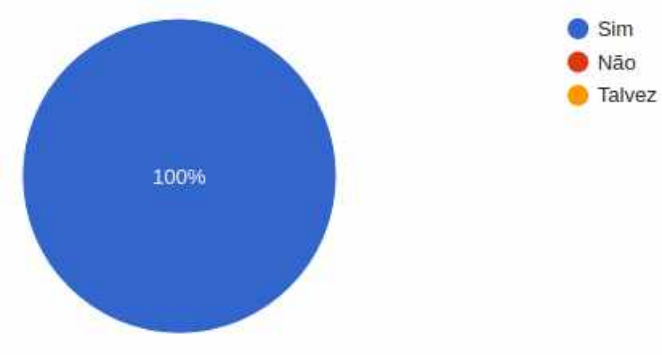

Figura 39 – Gráfico da questão 06 sobre a instalação da aplicação.

07) Você considera o aplicativo de fácil utilização?*
a) Sim
b) Não
c) Talvez

Resultado:

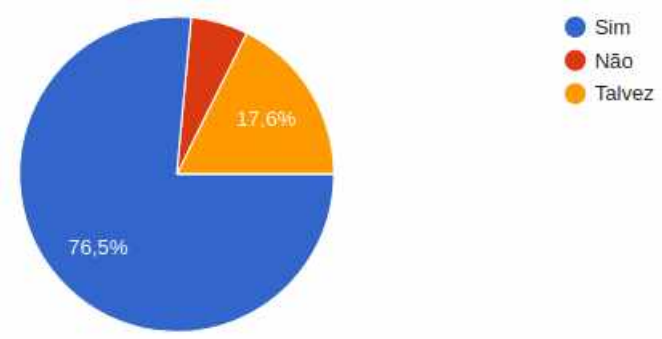

Figura 40 – Gráfico da questão 07 sobre a utilização da aplicação.

08) Sugestões ou comentários:

A ultima questão é aberta e não obrigatória para que o usuário deixe sua opinião ou sugestão de acordo com a experiencia de utilização da aplicação.

Comentários obtidos, transcritos literalmente:

1) Tive uma experiência bem agradável ao usá-lo, caso tivesse utilizado antes para realizar meu TCC teria diminuído bastante o tempo que levei para encontrar artigos que precisava, achei bem útil ao fazer testes nele, e também fácil de mexer e instalar. Quanto a sugestão, sugiro que futuramente adéque o programa para uma página Web, para se tornar mais prático e acessível para os usuários. Parabéns pela excelente ferramenta criada.

2) O sistema me surpreendeu, foi bem mais eficaz que os outros que já utilizei, gostei também da simplicidade, achei bem fácil a interação com ele, tenho certeza que irá ajudar os usuários a diminuir seu tempo de busca por um documento científico conforme ele deseja.

3) - Seria interessante ter uma interface.
- Com uma visão de usuário, seria possível reduzir o tempo de inicialização do aplicativo?
- Trocar todas as palavras "sinonimos"por "sinônimos", ou seja adicionar acento.
- Trocar a opção "1 - Fazer busca"por "1 - Realizar busca"
- Sugestão após seleciona a opção "2 - "é exibido uma frase que contem a palavra "presione", que na verdade se escreve "pressione". com dois "SS".
- Substituir a frase "Qual o idioma dos documentos desta coleção?"por "Qual é o idioma dos documentos contidos na coleção ?"
- No menu dentro da opção "3 - Gerenciar sinonimo"colocar acento em "Dicionário",.
- Trocar a frase "Nem um caminho selecionado."trocar por "Nenhum caminho selecionado."
- A lista de opções da aplicação em teste não resultou na mesma ordem que a busca do google retornou. Porém o primeiro resultado retornado pela aplicação de fato era o melhor conteúdo. Mas o terceiro conteúdo retornado da aplicação não foi muito relacionado com o tem que buscava, não sei se há relação, mas o arquivo possui muito a vezes a palavra principal. e muito pouco das demais palavras.
- Achei um pouco chato ter que baixar 10 documentações para serem analisadas. Porem acredito que esta aplicação não será utilizada da maneira que utilizei, e sim será utilizada com um integração em um repositório de arquivo, o que é bem legal.

4) O aplicativo no geral funcionou bem e é de fácil utilização, é necessário fazer alguns ajustes pois dependendo da pesquisa ele fecha.

5) Poderia aceitar documentos em inglês e português juntos

4.2 Análise dos resultados

Dos 17 questionários respondidos, obtivemos as seguintes respostas:

Questões

01) 100% consideram que um sistema que economiza seu tempo para encontrar documentos relevantes é importante.

02) 88,2% sofre com cansaço físico, visual ou dores nas costas ao passar muito tempo na frente do computador, 5,9% respondeu talvez e 5,9% respondeu que não sofre nem um tipo de cansaço.

03) 64,7% consideram que o resultado da aplicação foi melhor que o mecanismo e 35,3% consideram que pode ser melhor.

04) 70,6% consideram que o uso de sinônimos pela aplicação teve os resultados melhores que o mecanismo de busca e 29,4% consideram que pode ser melhor.

05) 76,5% consideram que o aplicativo pode reduzir o tempo gasto lendo documentos e 23,5% consideram que pode ser melhor.

06) 100% consideram o aplicativo de fácil instalação.

07) 76,5% consideram o aplicativo de fácil utilização, 17,6% consideram que talvez seja de fácil utilização e 5,9% não consideram a aplicação de fácil utilização.

08) Nos comentários deixados pelos usuários, nota-se que a aplicação pode diminuir o tempo de leitura de artigos e que os documentos podem ser ter uma melhor classificação que o mecanismo de busca clássico. Houveram algumas sugestões de melhorias da aplicação, como correção ortográfica do texto em geral, sugestão de uma interface gráfica para interação e a disponibilização de uma página Web e suporte a varias línguas. Outro ponto citado é uma das limitações da aplicação de não buscar os documentos, sendo necessário o usuário fornece-los e a possibilidade de classificação de documentos com idiomas diferentes dentro da mesma coleção, não sendo possível na versão disponibilizada por possuir a necessidade de traduzir um documento para que todos tenha o vocabulário no mesmo idioma. Um dos pontos citados pelos comentários, foi relatado que exite uma falha ao realizar busca mas não foi especificado em que momento ela acontece.

Analisando essas resposta obtidas, percebemos que as questões 1, 2 e 5, considera que os usuários têm interesse em um sistema que economizasse seu tempo, que a grande maioria dos usuários sofre com algum tipo de cansaço físico, visual ou dores ao passar muito tempo na frente do computador e que o uso do aplicativo pode reduzir o tempo gasto analisando documentos. Nas questões 3 e 4, considera que o sistema pode obter resultados iguais ou melhores que o mecanismo de busca utilizado pelo usuário. Nas questões 6 e 7, considera que a aplicação é de fácil instalação e utilização pelo usuário. Reafirmando os pontos propostos do trabalho sobre economia de tempo, melhoria da saúde do usuário e melhoria na qualidade do trabalho.

Capítulo 5

Conclusão

Embora os mecanismos de busca clássicos tenham uma grande capacidade de recuperar documentos, eles podem ser tendenciosos na forma de classificar os resultados obtidos. Vários fatores são utilizados na hora da busca para que possa otimizar a busca em milhares de documentos e trazer um resultado interessante para o usuário em um tempo razoável. Os resultados obtidos acabam sendo afetados por elementos fora do contexto dos próprios documentos, podendo impactar negativamente na qualidade das buscas. Este impacto nas buscas pode ser refletido em um conjunto maior de documentos retornados ao usuário que não são interessantes a ele, levando a um maior gasto de tempo em análise de documentos não relevantes.

Este trabalho propôs o desenvolvimento de uma aplicação que classifique por relevância uma coleção de documentos de um repositório, em formato PDF ou TXT. O processo proposto extrai o conteúdo textual dos documentos, aplica várias técnicas de pré-processamento de Recuperação de Informação, modela na estrutura *Bag-Of-Words*, aplica o modelo Vetorial com as métricas TF e IDF. A esta aplicação foi adicionado um dicionário de sinônimos a ser preenchido pelo usuário. No momento das buscas a aplicação expande a busca baseada no dicionário. Esta aplicação não considera os fatores externos ao conteúdo dos documentos e busca reduzir o tempo de pesquisa dos usuários por meio de uma classificação mais eficiente.

Com a análise dos resultados obtidos pelo questionário, pode-se concluir que a maioria dos usuários sofre de algum tipo de dor ou sintoma ao ficar muito tempo na frente do computador e que um sistema que auxilia e melhora os resultados de um mecanismo de busca clássico é de interesse dos usuários. Foi constatado que ao utilizar o conteúdo textual dos documentos unido a expansão de consulta, obtém-se resultados tão bons ou melhores se comparado a um mecanismo de busca clássico que contem fatores externos.

A grande maioria dos usuários acredita que a aplicação desenvolvida cumpre o papel de auxiliar nas buscas melhorando a qualidade da classificação e diminuindo o número de documentos a ser analisado. A grande maioria também acredita que a aplicação pode reduzir o tempo gasto na frente do computador.

Assim, com a avaliação dos usuários, pode-se concluir que a aplicação desenvolvida neste trabalho atingiu seu objetivo.

5.1 Trabalhos Futuros

- Implementação de uma interface gráfica.

- Implementação de um serviço web para desenvolvimento de uma aplicação web e mobile.

- Implementação de um dicionário genérico de sinônimos.

- Busca dos documentos públicos na web, sem a necessidade de o usuário fornecer tais documentos.

- Expandir os idiomas suportados.

- Suportar coleções com documentos em mais de um idioma.

- Realizar um estudo e sua possível implementação de outros métodos.

- Realizar um estudo e sua possível de outros métodos de expansão.

5.2 Ações futuras

- Estudar e disponibilizar o aplicativo em algum repositório de acesso da universidade.

Referências

About Google. **Da garagem para o Googleplex**. 2019. Disponível em: <https://about.google/our-story>. Acesso em: 26 de Setembro de 2019. Citado na página 11.

AFUAN AHMAD ASHARI, Y. S. L. **A study: query expansion methods in information retrieval**. 2019. Disponível em: <https://iopscience.iop.org/article/10.1088/1742-6596/1367/1/012001/pdf>. Acesso em: 08 de Dezembro de 2020. Citado na página 26.

Alison J. Head. **Beyond Google: How do students conduct academic research**. 2007. Disponível em: <https://firstmonday.org/article/view/1998/1873>. Acesso em: 26 de Setembro de 2019. Citado na página 11.

ANALYTICS, O. **What is Inverse Document Frequency**. 2019. Disponível em: <https://www.opinosis-analytics.com/knowledge-base/inverse-document-frequency-idf-explained/>. Acesso em: 05 de Fevereiro de 2020. Citado na página 22.

_____. **What is Term Frequency**. 2019. Disponível em: <https://www.opinosis-analytics.com/knowledge-base/term-frequency-explained/>. Acesso em: 05 de Fevereiro de 2020. Citado na página 22.

Araújo Júnior, R. H. d. Precisão no processo de busca e recuperação da informação. 2005. Disponível em: <http://repositorio.unb.br/handle/10482/34608>. Acesso em: 29 de Outubro de 2019. Citado na página 11.

Baeza-Yates, B. R.-N. R. **Modern Information Retrieval: The Concepts and Technology Behind Search**. 2. ed. Addison Wesley, 2010. ISBN 0321416910,9780321416919. Disponível em: <http://gen.lib.rus.ec/book/index.php?md5=3441281814aa5fa0411d58d6d1510c9b>. Acesso em: 18 de Desembro de 2019. Citado 2 vezes nas páginas 18 e 24.

Barros, S. S. de; Ângelo, R. d. C. de O.; Uchôa, É. P. B. L. Lombalgia ocupacional e a postura sentada. SciELO Brasil, 2011. Disponível em: <http://www.scielo.br/pdf/rdor/v12n3/v12n3a06>. Acesso em: 16 de Dezembro de 2019. Citado na página 14.

BBC, B. **Uso de tablets ou computador prejudica sono, diz estudo**. 2015. Disponível em: <ttps://www.bbc.com/portuguese/noticias/2015/02/150205_telas_tablet_sono_fn>. Acesso em: 16 de Dezembro de 2019. Citado na página 14.

Beel, J.; Gipp, B. **Google Scholar's ranking algorithm: the impact of citation counts (an empirical study)**. 2009. Disponível em: <https://ieeexplore-ieee-org.ez34.periodicos.capes.gov.br/abstract/document/5089308>. Acesso em: 16 de Dezembro de 2019. Citado 2 vezes nas páginas 12 e 13.

Bing. **Bing**. 2009. Disponível em: <https://www.bing.com/?c=br>. Acesso em: 29 de Outubro de 2019. Citado 2 vezes nas páginas 12 e 16.

Blehm, C. et al. Computer vision syndrome: A review. **Survey of Ophthalmology**, v. 50, n. 3, p. 253 – 262, 2005. ISSN 0039-6257. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0039625705000093>. Citado 2 vezes nas páginas 13 e 14.

BROWNLEE, J. **A Gentle Introduction to the Bag-of-Words Model**. 2017. Disponível em: <https://machinelearningmastery.com/gentle-introduction-bag-words-model/>. Acesso em: 05 de Fevereiro de 2020. Citado na página 20.

Cambridge University Press. **A first take at building an inverted index**. 2009. Disponível em: <https://nlp.stanford.edu/IR-book/html/htmledition/a-first-take-at-building-an-inverted-index-1.html>. Acesso em: 05 de Fevereiro de 2020. Citado na página 21.

Carpineto, C.; Romano, G. A survey of automatic query expansion in information retrieval. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 44, n. 1, p. 1–50, jan. 2012. ISSN 0360-0300. Disponível em: <http://doi-acm-org.ez34.periodicos.capes.gov.br/10.1145/2071389.2071390>. Acesso em: 06 de Outubro de 2019. Citado na página 12.

Castells, P.; Fernandez, M.; Vallet, D. An adaptation of the vector-space model for ontology-based information retrieval. **IEEE Transactions on Knowledge and Data Engineering**, v. 19, n. 2, p. 261–272, Feb 2007. ISSN 2326-3865. Disponível em: <https://ieeexplore.ieee.org/document/4039288>. Acesso em: 30 de Dezembro de 2019. Citado na página 26.

Cornell University Library. **Term Weighting Approaches in Automatic Text Retrieval**. 1987. Disponível em: <https://ecommons.cornell.edu/bitstream/handle/1813/6721/87-881.pdf?sequence=1{&}isAllowe>. Acesso em: 05 de Fevereiro de 2020. Citado na página 22.

CROFT, W. B. Document representation in probabilistic models of information retrieval. **Journal of the American Society for Information Science**, Wiley Online Library, v. 32, n. 6, p. 451–457, 1981. Disponível em: <https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.4630320609>. Citado na página 25.

DEAN, B. **Google's 200 Ranking Factors: The Complete List**. 2020. Disponível em: <https://backlinko.com/google-ranking-factors>. Acesso em: 04 de Fevereiro de 2020. Citado na página 12.

Flores, F. N.; Moreira, V. P. Assessing the impact of stemming accuracy on information retrieval - a multilingual perspective. **Information Processing & Management**, 2016. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0306457316300358>. Citado na página 19.

GARCIA, E. Robertson-spärck-jones probabilistic model tutorial. 03 2009. Acesso em: 30 de Setembro de 2019. Citado na página 25.

Gomes, G. M. R.; Cendón, B. V. Análise da integração da recuperação da informação, information search behaviour e interação humano-computador para avaliação de sistemas de recuperação da informação. **Transinformação**, Pontifícia Universidade Católica de Campinas, v. 27, n. 3, p. 277–284, 2015. Acesso em: 18 de Desembro de 2019. Citado na página 16.

Google. **Google**. 1998. Disponível em: <https://www.google.com.br>. Acesso em: 29 de Outubro de 2019. Citado 4 vezes nas páginas 11, 12, 16 e 27.

GOOGLE. **Introdução ao robots.txt**. 2018. Disponível em: <https://support.google.com/webmasters/answer/6062608?hl=pt-BR>. Acesso em: 04 de Fevereiro de 2020. Citado na página 19.

Google Acadêmico. **Google Acadêmico**. 2004. Disponível em: <https://scholar.google.com.br/scholar>. Acesso em: 29 de Outubro de 2019. Citado na página 12.

Google Trends. **Veja o que o mundo está pesquisando**. 2006. Disponível em: <https://trends.google.com.br/trends/?geo=BR>. Acesso em: 29 de Outubro de 2019. Citado na página 12.

Google Webmaster Central Blog. **PDFs in Google search results**. 2011. Disponível em: <https://webmasters.googleblog.com/2011/09/pdfs-in-google-search-results.html>. Acesso em: 06 de Outubro de 2019. Citado na página 12.

_____. **Identifying valid search engine ranking factors in a Web 2.0 and Web 3.0 context for building efficient SEO mechanisms**. 2015. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0952197615000329>. Acesso em: 06 de Novembro de 2019. Citado na página 12.

Jahn, G. F. **Uma proposta de arquitetura para tratamento de dados não estruturados no âmbito dos institutos federais de educação**. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2017. Acesso em: 18 de Desembro de 2019. Citado na página 16.

Lashkari, A. H.; Mahdavi, F.; Ghomi, V. **A Boolean Model in Information Retrieval for Search Engines**. 2009. Disponível em: <https://ieeexplore.ieee.org/abstract/document/5077062>. Acesso em: 05 de Fevereiro de 2020. Citado na página 24.

MANNING, P. R. C. D.; SCHüTZE, H. **Introduction to Information Retrieval**. 2008. Disponível em: <https://nlp.stanford.edu/IR-book/pdf/09expand.pdf>. Acesso em: 08 de Dezembro de 2020. Citado na página 26.

Monteiro, S. D. et al. Sistemas de recuperação da informação e o conceito de relevância nos mecanismos de busca: semântica e significação. **Encontros Bibli: revista eletrônica de biblioteconomia e ciência da informação**, v. 22, n. 50, p. 161–175, 2017. Acesso em: 18 de Desembro de 2019. Citado na página 16.

MORAES, D. **Web crawler: saiba o que é e qual a sua relação com o Marketing Digital**. 2018. Disponível em: <https://rockcontent.com/blog/web-crawler/>. Acesso em: 04 de Fevereiro de 2020. Citado na página 19.

Orduna-Malea, E. et al. **About the size of Google Scholar: playing the numbers**. 2014. Disponível em: <https://arxiv.org/abs/1407.6239>. Acesso em: 16 de Dezembro de 2019. Citado na página 13.

PURBASARI, I.; ANGGRAENY, F.; MAHARANI, M. **Replies Identification in Question Answering System using Vector Space Model**. 2018. Acesso em: 30 de Novembro de 2020. Citado na página 27.

RAMOS, J. et al. **Using tf-idf to determine word relevance in document queries**. 2003. Acesso em: 05 de Fevereiro de 2020. Citado na página 23.

SAIF, H. et al. **On stopwords, filtering and data sparsity for sentiment analysis of Twitter**. 2014. Disponível em: <http://oro.open.ac.uk/40666/>. Citado na página 19.

SALTON, G. **The Smart environment for retrieval system evaluation—advantages and problem areas**. 2008. Disponível em: <https://sigir.org/files/museum/Information_Retrieval_Experiment/pdfs/p316-salton.pdf>. Acesso em: 08 de Dezembro de 2020. Citado na página 26.

Sanderson, M.; Croft, W. B. The history of information retrieval research. **Proceedings of the IEEE**, v. 100, n. Special Centennial Issue, p. 1444–1451, May 2012. Acesso em: 26 de Setembro de 2019. Citado na página 11.

SARKAR, I. N. A vector space model approach to identify genetically related diseases. **Journal of the American Medical Informatics Association**, v. 19, n. 2, p. 249–254, 01 2012. ISSN 1067-5027. Disponível em: <https://doi.org/10.1136/amiajnl-2011-000480>. Acesso em: 30 de Dezembro de 2019. Citado na página 26.

SINGH, J. N.; DWIVEDI, S. K. Performance evaluation of search engines using enhanced vector space model. **Journal of Computer Science**, Citeseer, v. 11, n. 4, p. 692, 2015. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi\unhbox\voidb@x\bgroup\let\unhbox\voidb@x\setbox\@tempboxa\hbox{1\global\mathchardef\accent@spacefactor\spacefactor}\accent91\egroup\spacefactor\accent@spacefactor0.1.1.967.4337{&}rep=rep1{&}ty>. Acesso em: 30 de Dezembro de 2019. Citado na página 27.

SINGH, P. **Fundamentals of Bag Of Words and TF-IDF**. 2019. Acesso em: 08 de Dezembro de 2020. Citado na página 20.

Support Google. **Como a Pesquisa Google funciona**. 2019. Disponível em: <https://support.google.com/webmasters/answer/70897>. Acesso em: 26 de Setembro de 2019. Citado na página 11.

TUNKELANG, D. **Query Expansion**. 2017. Disponível em: <https://queryunderstanding.com/query-expansion-2d68d47cf9c8>. Acesso em: 08 de Dezembro de 2020. Citado na página 25.

WONG, S. M.; RAGHAVAN, V. V. Vector space model of information retrieval: a reevaluation. 1984. Disponível em: <https://www.researchgate.net/profile/Vijay_Raghavan10/publication/221300847_Vector_Space_Model_of_Information_Retrieval_-_A_Reevaluation/links/00b4952d294dea632e000000/

Vector-Space-Model-of-Information-Retrieval-A-Reevaluation.pdf>. Acesso em: 05 de Fevereiro de 2020. Citado na página 25.

XIA, Y. et al. **Joint Inverted Indexing**. 2013. Disponível em: <http://openaccess.thecvf.com/content_iccv_2013/papers/Xia_Joint_Inverted_Indexing_2013_ICCV_paper.pdf>. Citado na página 21.

Yahoo. **Yahoo**. 2019. Disponível em: <https://br.yahoo.com>. Acesso em: 26 de Setembro de 2019. Citado 3 vezes nas páginas 11, 12 e 27.

Yates, R. B.; Neto, B. R. **Recuperação de Informação-: Conceitos e Tecnologia das Máquinas de Busca**. [S.l.]: Bookman Editora, 2013. Acesso em: 31 de Outubro de 2019. Citado na página 11.