UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

MARCOS FELIPE PONTES REZENDE
Orientador: Guilherme Tavares de Assis

ATRI: UM AMBIENTE EXPERIMENTAL DE RECUPERAÇÃO DE INFORMAÇÃO

Ouro Preto, MG
2022

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

MARCOS FELIPE PONTES REZENDE

ATRI: UM AMBIENTE EXPERIMENTAL DE RECUPERAÇÃO DE INFORMAÇÃO

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Guilherme Tavares de Assis

Ouro Preto, MG
2022

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

R467a Rezende, Marcos Felipe Pontes.
ATRI [manuscrito]: um ambiente experimental de Recuperação de Informação. / Marcos Felipe Pontes Rezende. - 2022.
65 f.: il.: color., gráf., tab..

Orientador: Prof. Dr. Guilherme Tavares de Assis.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Biológicas. Graduação em Ciência da Computação .

1. Recuperação da Informação. 2. Algoritmos. 3. Cálculo de Similaridade. I. de Assis, Guilherme Tavares. II. Universidade Federal de Ouro Preto. III. Título.

CDU 004

Bibliotecário(a) Responsável: Luciana De Oliveira - SIAPE: 1.937.800

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E BIOLOGICAS
DEPARTAMENTO DE COMPUTACAO

FOLHA DE APROVAÇÃO

Marcos Felipe Pontes Rezende

ATRI: Um ambiente experimental de recuperação de informação

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal
de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 14 de Junho de 2022.

Membros da banca

Guilherme Tavares de Assis (Orientador) - Doutor - Universidade Federal de Ouro Preto
Rodrigo Geraldo Ribeiro (Examinador) - Doutor - Universidade Federal de Ouro Preto
Rodrigo César Pedrosa Silva (Examinador) - Doutor - Universidade Federal de Ouro Preto

Guilherme Tavares de Assis, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 14/06/2022.

Documento assinado eletronicamente por **Guilherme Tavares de Assis**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 14/06/2022, às 20:14, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0343225** e o código CRC **CBB2C0FC**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: 3135591692 - www.ufop.br

Dedico este trabalho a todos que me incentivaram e me ajudaram para que isso acontecesse, em especial aos meus pais e minha namorada.

Agradecimentos

Gostaria de agradecer a todos que me ajudaram de alguma forma durante minha graduação. Em especial, faço um agradecimento aos meus pais Rosilene e Osmar por terem me incentivado e apoiado sempre com paciência e confiança.

Gostaria de agradecer também à minha namorada, Letícia, que está ao meu lado desde o início da graduação e foi fundamental para tornar todo este processo mais leve, tranquilo e divertido.

Agradeço também ao meu orientador Guilherme Tavares, que me ensinou muito durante todos estes anos com bom humor, confiança e excelência.

Aos demais professores do DECOM, deixo meu sincero agradecimento. Com certeza levarei os ensinamentos para toda a vida.

Para finalizar, agradeço aos amigos que estiveram presentes ao longo dessa caminhada. Com certeza, todos foram essenciais em algum sentido e levarei as lembranças para toda a vida.

"Rasgue o céu que der."(ESPINHAÇO, 2018)

Resumo

Dentro da área de Recuperação de Informação (RI), algoritmos de ranqueamento são responsáveis por decidir, mediante uma consulta, quais documentos são relevantes ou não à mesma. Neste contexto, visando garantir eficácia aos resultados retornados, é necessário uma modelagem apropriada dos documentos da coleção e das consultas consideradas, no intuito de produzir, adequadamente, uma função de ranqueamento que atribui pontuações de similaridade entre uma consulta e documentos de uma coleção. Para definição de funções de ranqueamento, diversos modelos de RI foram propostos, utilizando-se de formulações booleanas, vetoriais e probabilísticas. Cada modelo de RI possui suas próprias premissas de funcionamento que levam à classificação de documentos de uma determinada coleção, mediante consultas desejadas. Desta forma, este trabalho tem, como objetivo principal, a proposta, o desenvolvimento e a validação de um ambiente experimental de RI, denominado ATRI, que se baseia em distintos modelos de RI para cálculo de similaridade entre consultas e documentos de uma coleção por meio de uma interface amigável, podendo ser aplicado em situações diversas. Para tanto, foram considerados, para cálculo de similaridade, os modelos Booleano, Vetorial, Probabilístico, BM25, Redes de Crença, Booleano Estendido, Vetorial Generalizado, DFRee, PL2 e PageRank. Além disso, o ATRI permite a criação de um ambiente de avaliação de eficácia e *benchmarking* para a área de RI, por meio da criação automática de *ensembles*, visualização de métricas de eficácia e suporte a coleções textuais diversas. Para validar o ambiente proposto e desenvolvido, experimentos foram realizados, envolvendo quatro coleções de teste padronizadas e suas respectivas consultas. Por meio da análise dos resultados dos experimentos realizados, foi possível perceber que o modelo BM25 apresentou os melhores resultados quando comparado aos demais modelos de RI, embora todos tenham apresentados resultados satisfatórios em contextos particulares, e que a utilização de *ensembles*, por combinar boas características dos modelos suportados pelo ATRI, pode ser capaz de criar funções de ranqueamento mais eficazes.

Palavras-chave: Algoritmos de Ranqueamento. Cálculo de similaridade. Modelos clássicos de recuperação de informação. Modelos estendidos de recuperação de informação. Criação automática de *ensembles*.

Abstract

In Information Retrieval (IR), ranking algorithms are responsible for deciding, based on a query, which documents are relevant to it. In this context, to guarantee the effectiveness of the returned results, an appropriate modeling of the considered documents and queries is necessary, aiming to produce ranking functions that assigns similarity scores between a query and documents from a collection. To define ranking functions, several IR models were proposed, using boolean, vectorial and probabilistic formulations. Each IR model has its foundations that lead to the rank of documents from a given corpus based on different queries. Thus, this work has, as main objective, the proposal, development, and validation of an experimental IR environment, called ATRI, which includes different IR models to calculate the similarity between queries and documents in a collection through a friendly interface, and could have applications in different scenarios. For this purpose, the following models were considered for similarity calculation: Boolean, Vector Space, Probabilistic, BM25, Belief Network, Extended Boolean, Generalized Vector Space, DFRee, PL2, and PageRank. In addition, ATRI allows the creation of a benchmarking environment for evaluation of the effectiveness and performance on IR through automatic creation of ensembles, visualization of effectiveness metrics, and support to distinct collections. In order to validate the proposed environment developed, experiments were performed involving four standardized test collections and their respective queries. Analyzing the experiments' results, it was possible to see that the BM25 model presented better results when compared to the other tested IR models, despite all models presenting satisfactory results for particular contexts, and also that the usage of ensembles, that combines good of all models supported by ATRI, may be qualified to create more efficient ranking functions.

Keywords: Ranking Algorithms. Similarity calculation. Classic models of information retrieval. Extended models of information retrieval. Automatic creation of ensembles.

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

AM	Aprendizado de Máquina
BC	*Borda Count*
BM	*Boolean Model*
BM25	*Best Match 25*
BNM	*Belief Network Model*
CF	*Cystic Fibrosis*
DCG	*Discounted Cumulative Gain*
DFR	*Divergence From Randomness*
DFRee	*DFR Free of Parameters*
EBM	*Extended Boolean Model*
GVSM	*Generalized Vector Space Model*
IA	Inteligência Artificial
IDF	*Inverse Document Frequency*
LISA	*Library Information Science Collection*
LTR	*Learning To Rank*
MAP	*Mean Average Precision*
MC	*Markov Chain*
NDCG	*Normalized Discounted Cumulative Gain*
PM	*Probabilistic Model*
RI	Recuperação de Informação
TF	*Term Frequency*
VSM	*Vector Space Model*

Lista de Símbolos

$\in$	Pertence
$\exists$	Existe
$\cap$	Interseção
$\cup$	União
$\wedge$	*AND* Lógico
∞	Infinito
$\forall$	Para todo
λ	Lambda
$\succ$	Mais relevante que
π	Pi

Sumário

1 Introdução

Por mais de 5000 anos, de acordo com (BAEZA-YATES; RIBEIRO-NETO, 2013), a humanidade vem organizando a informação para posterior busca e recuperação. Em sua forma mais usual, isto foi feito utilizando itens textuais (documentos) que, por sua vez, foram sendo organizados ao longo da história em grandes bibliotecas. Com a evolução da Web, no entanto, o volume destes itens de informação cresce rapidamente e sem obedecer qualquer tipo de organização; desta forma, estes itens são disponibilizados de maneira livre e desorganizada na Web. Tal desorganização levou ao crescimento de uma área da Ciência da Computação, denominada Recuperação de Informação (RI). A RI visa, de uma forma geral, prover aos usuários acesso fácil e eficaz às informações de seu interesse (ALVAREZ; GONÇALVES, 2017).

De acordo com (WIVES, 1997), a RI refere-se ao ato do usuário especificar e descrever a informação de que ele precisa, juntamente com as técnicas utilizadas para recuperar essas informações. Para tanto, a RI lida com a representação, o armazenamento, a organização e o acesso a itens de informação. Neste sentido, segundo (GROSSMAN; FRIEDER, 2012), a RI é uma área dedicada à procura de documentos relevantes, de acordo com a necessidade ou o interesse de usuários, em uma coleção de documentos, e não a simplesmente detectar casamentos de um padrão desejado em documentos.

Entretanto, não é uma tarefa fácil determinar o que é realmente relevante para os usuários, visto que um usuário pode não possuir uma descrição bem detalhada sobre seu objeto de consulta. Segundo (CAMBAZOGLU; BAEZA-YATES, 2016), uma simples consulta de um usuário por meio de uma máquina de busca, por exemplo, pode apresentar um número elevado de referências, como resultado, que não atendem ao contexto da consulta realizada. Desta forma, visando garantir eficácia aos resultados retornados, é necessário uma modelagem apropriada dos documentos da coleção e das consultas, no intuito de produzir, adequadamente, uma função de similaridade que atribui pontuações eficazes de similaridade a documentos da coleção em relação à consulta, especificada em linguagem natural (ALVAREZ; GONÇALVES, 2017).

As funções de similaridade, segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), são responsáveis por decidir, mediante uma consulta, quais documentos são relevantes ou não à mesma, estabelecendo uma ordem para os resultados retornados. Para definição de funções de similaridade, inicialmente, os seguintes modelos de RI, denominados clássicos, foram propostos: o Booleano, o Vetorial e o Probabilístico. De uma forma geral, o modelo Booleano utiliza-se de um arcabouço teórico baseado na teoria de conjuntos, representando documentos e consultas como conjuntos de termos de indexação[1]. Já no modelo Vetorial, os documentos e consultas são representados como vetores, categorizando-se como um modelo algébrico. Por fim, o modelo

[1] De acordo com (BAEZA-YATES; RIBEIRO-NETO, 2013), um termo de indexação é uma palavra ou um grupo de palavras consecutivas que representam conceitos-chave (ou tópicos) em um documento.

Probabilístico utiliza-se da teoria das probabilidades para definir sua função de similaridade.

A partir dos modelos clássicos de RI, diversos outros modelos, denominados estendidos, foram propostos. O modelo Booleano Estendido, por exemplo, foi proposto como uma extensão do modelo clássico Booleano, adicionando-se representações vetoriais propostas pelo modelo Vetorial. Outras extensões foram propostas baseadas somente no modelo Vetorial, como o modelo Vetorial Generalizado, que estende o modelo clássico por meio da atribuição de correlações entre os termos de indexação. Ademais, diversos outros modelos foram criados a partir da formulação probabilística clássica, como o de Redes de crença, BM25, DFRee e PL2. Por fim, outros modelos alternativos também foram propostos baseando-se puramente na estruturação dos documentos da coleção, como o famoso PageRank. Segundo (KRAAIJ, 2004), o modelo Booleano Estendido não é usual porque a montagem das consultas é complexa, embora a utilização de interfaces de apoio juntamente com consultas curtas poderia torná-lo um modelo atrativo. O Vetorial Generalizado, por sua complexidade computacional, só é aplicável para consultas curtas em uma pequena quantidade de documentos. Os modelos probabilísticos, por sua vez, têm seu comportamento ditado pela estratégia de ranqueamento escolhida, podendo ser úteis em situações diversas. Já o PageRank só pode ser utilizado em documentos semi-estruturados[2]. Dessa forma, cada um dos modelos estendidos possui diferentes ganhos em relação aos clássicos e pode ser utilizado em situações específicas.

Recentemente, devido à esta vasta quantidade de modelos distintos e à existência de dados com potencial de treinamento disponíveis, tornou-se possível aproveitar as tecnologias existentes de Aprendizado de Máquina (AM) e RI para construir modelos de ranqueamento ainda mais eficazes. Em linhas gerais, AM é uma subárea da Inteligência Artificial (IA) cujo principal objetivo está relacionado ao desenvolvimento de técnicas computacionais para o aprendizado e construção de sistemas capazes de adquirir conhecimento de forma automática (MONARD; BARANAUSKAS, 2003). Especificamente, métodos que aprendem como combinar *features* predefinidas para ranqueamento de itens por meio de AM são chamados de métodos *Learning To Rank* (LTR). Aplicada no contexto de RI, a utilização de AM é motivada por inúmeros fatores. De acordo com (LI, 2011), é uma escolha natural, atualmente, incorporar os *scores* gerados por algoritmos já existentes, além de informações de *feedback* fornecidos pelos usuários tais como os inúmeros registros de clique em determinados documentos retornados, como parâmetros para criar um modelo de ranqueamento. Além disso, segundo (LI, 2014), também é possível utilizar técnicas de AM não supervisionadas para agregação de ranqueamentos, no intuito de produzir modelos *ensemble*[3] de forma automática. De fato, a utilização de LTR para recuperação de documentos tem se tornado uma das técnicas chave no contexto moderno de busca, especialmente na Web.

[2] Dados semi-estruturados apresentam uma representação estrutural heterogênea, não sendo nem completamente não-estruturados nem estritamente tipados (MELLO et al.,). Um exemplo comum são os documentos baseados em hipertexto: composto por páginas e ligações entre elas.

[3] Em estatística e AM, os métodos *ensemble* combinam vários algoritmos para obter melhor desempenho preditivo do que o obtido por qualquer um dos algoritmos individualmente.

Este capítulo encontra-se organizado como se segue. A Seção 1.1 apresenta a motivação para a realização desse trabalho. A Seção 1.2 descreve os objetivos geral e específicos. A Seção 1.3 aborda o método utilizado no desenvolvimento desse trabalho. Finalmente, a Seção 1.4 apresenta o delineamento do restante da monografia.

1.1 Justificativa

Atualmente, não existe algum ambiente disponível que se baseia em distintos modelos de RI para cálculo de similaridade entre consultas e documentos de uma coleção por meio de uma interface amigável, o que pode ser aplicado em situações diversas. Este trabalho propõe o ambiente ATRI, que pode ser aplicado em situações em que se deseja saber a proximidade de um determinado documento ou um conjunto de termos de interesse do usuário em relação a um conjunto de documentos pertencentes a uma coleção, automatizando tal processo.

Uma ferramenta similar para cálculo de similaridade entre documentos foi desenvolvida por (JANEIRO, 2017). A ferramenta proposta contemplou os modelos clássicos Booleano, Vetorial e Probabilístico e o modelo Booleano Estendido e apresentou resultados experimentais satisfatórios para pequenas coleções. No entanto, a proposta trata apenas destes modelos de RI, limitando-se a técnicas mais clássicas de recuperação e a pequenos conjuntos de dados. Logo, há necessidade de se desenvolver um ambiente mais robusto que englobe mais modelos, além da possibilidade de customização por meio da criação de *ensembles* e da obtenção de métricas de eficácia.

1.2 Objetivos

Este trabalho possui, como objetivo geral, a proposta e o desenvolvimento de um ambiente de RI, denominado ATRI, para cálculo de similaridade entre consultas e coleções de documentos, possibilitando a geração de ranqueamentos de relevância para cada consulta desejada, que pode ser um determinado documento ou um conjunto de termos de interesse do usuário. Para tanto, o ATRI permite a utilização dos seguintes modelos de RI como base para o cálculo de similaridade: Booleano, Vetorial, Probabilístico, BM25, Redes de Crença, Booleano Estendido, Vetorial Generalizado, DFRee, PL2 e PageRank.

Além disso, o ATRI permite a criação de um ambiente de avaliação de eficácia e *benchmarking* para a área de RI, por meio da criação automática de *ensembles*, visualização de métricas de eficácia e suporte à diferentes coleções científicas, possibilitando ter uma grande visibilidade acadêmica. Desta forma, o ambiente também pode ser útil para pesquisas científicas em RI cujo objetivo envolve a criação e aprimoramento de funções de ranqueamento personalizadas, além da avaliação das mesmas por meio da coleta de métricas de eficácia em coleções preestabelecias.

De um modo geral, os principais objetivos específicos, alcançados neste trabalho, foram:

- geração de métricas de eficácia e *benchmarking* para consultas considerando distintos modelos de RI e coleções utilizadas;
- suporte a documentos de diferentes formatos textuais (não estruturados ou semi-estruturados) por meio da definição de esquemas de indexação, armazenamento e adequação destes itens textuais;
- suporte a distintos esquemas de ponderação aplicados aos modelos de RI implementados no ATRI;
- facilidade de parametrização para as configurações do ambiente, de forma a tornar personalizáveis tarefas como criação de coleções, indexação e configuração das funções de ranqueamento;
- obtenção de *feedback* do usuário quanto ao retorno de consultas realizadas, no intuito de gerar, automaticamente, métricas a respeito da eficácia das mesmas;
- suporte à criação automática de *ensembles*, no intuito de se construir funções de ranqueamento ainda mais eficazes por meio da combinação dos diferentes modelos presentes no ambiente.

1.3 Método de Trabalho

Visando o alcance do objetivo geral deste trabalho, experimentos foram realizados utilizando uma primeira versão finalizada do ambiente ATRI. O funcionamento do ambiente foi avaliado por meio da comparação empírica entre distintas funções de ranqueamento em diferentes coleções de teste. Quanto às coleções, foram definidas, inicialmente, três conjuntos de dados públicos e altamente reconhecidos: CF (*Cystic Fibrosis*), NPL e LISA (*Library Information Science Collection*). Para cada coleção, um conjunto de consultas pré-definidas com seus respectivos gabaritos também foram especificados. Ademais, uma nova coleção sobre o tema COVID19 foi definida para testar um cenário real de utilização do ambiente e avaliar o modelo PageRank. Por fim, é importante ressaltar que todas as funções de ranqueamento avaliadas foram construídas utilizando um ou mais modelos de RI presentes no ambiente, visando comparar a eficácia de funções de ranqueamento de modelos de RI únicos com funções de ranqueamento de modelos *ensemble*, compostos por distintos modelos de RI.

Para cada experimento, isto é, a realização de um tipo de consulta em uma determinada coleção e considerando uma função de ranqueamento, foi medida a eficácia de tal modelo de RI por meio das métricas de precisão e NDCG. Assim, a partir de tais experimentos, foi feita uma avaliação comparativa entre os modelos de RI presentes no ambiente.

1.4 Organização do Trabalho

O restante desta monografia encontra-se organizado como se segue. O Capítulo 2 apresenta a revisão de literatura necessária para a realização deste trabalho, envolvendo fundamentação teórica e trabalhos diretamente relacionados. O Capítulo 3 descreve o desenvolvimento do ATRI, envolvendo tópicos como arquitetura de funcionamento e interface. O Capítulo 4 descreve experimentos envolvendo os modelos de RI presentes no ATRI e analisa os resultados obtidos. Por fim, o Capítulo 5 apresenta as conclusões deste trabalho e as perspectivas de trabalho futuro.

2 Revisão de Literatura

Este capítulo apresenta a revisão de literatura feita para a realização deste trabalho. Para tanto, encontra-se organizado da seguinte maneira: a Seção 2.1 aborda a fundamentação teórica necessária ao desenvolvimento deste trabalho e a Seção 2.2 apresenta os trabalhos diretamente relacionados.

2.1 Fundamentação Teórica

Nesta seção, é apresentado o suporte teórico necessário para o entendimento e o desenvolvimento deste trabalho. A Subseção 2.1.1 refere-se ao sistema de RI e à tipologia básica dos modelos da área. As Subseções 2.1.2 e 2.1.3 tratam dos modelos clássicos e estendidos de RI, respectivamente. A Subseção 2.1.4 refere-se aos métodos de agregação de ranqueamento utilizados neste trabalho. Por fim, a Subseção 2.1.5 define uma estrutura de dados utilizada para indexar coleções: o Arquivo Invertido.

2.1.1 Sistemas e Modelos de RI

O propósito geral de um sistema de RI é ajudar os usuários a encontrar informações de seu interesse. Segundo (FERNEDA, 2003), os sistemas de RI devem representar o conteúdo dos documentos e apresentá-los ao usuário de uma maneira que lhe permita uma rápida seleção dos itens que satisfaçam total ou parcialmente à sua necessidade de informação, formalizada por meio de uma expressão de busca. Assim sendo, o principal objetivo de um sistema de RI é atingir alta eficácia quanto à proporção de satisfação versus o esforço do usuário.

De acordo com (BAEZA-YATES; RIBEIRO-NETO, 2013), um sistema em RI consiste em um sistema que, a partir de uma coleção de documentos, realiza a indexação destes documentos, a recuperação dos documentos relevantes da coleção mediante uma consulta do usuário e o ranqueamento dos documentos recuperados. Pode-se notar, desta forma, a dificuldade para obtenção de uma resposta qualitativa por um sistema de RI, dado a existência do fator humano: um documento pode ser considerado relevante para um determinado usuário e para outro não. Desta forma, um sistema de RI deve procurar sempre apresentar, como melhores resultados, aqueles documentos que podem ser considerados relevantes para uma maior quantidade de usuários (JANEIRO, 2017).

Neste sentido, os sistemas de RI possuem, em sua composição, um modelo de RI que, segundo (TURTLE; CROFT, 1992), é responsável por especificar as representações dos documentos e das consultas, além da função de similaridade, pela qual documentos e consultas são comparados por meio de um valor numérico. Uma possível caracterização formal para este mo-

delo descrito foi proposta por (BAEZA-YATES; RIBEIRO-NETO, 2013), na forma da quádrupla [**D**, **Q**, $\mathcal{F}$, **R**(q_i, d_j)] onde:

- **D** é um conjunto composto por visões lógicas (ou representações) dos documentos da coleção;
- **Q** é um conjunto composto por visões lógicas (ou representações) das necessidades de informação dos usuários;
- $\mathcal{F}$ é um arcabouço para modelar as representações dos documentos, das consultas e de seus relacionamentos, como conjuntos e relações booleanas, vetores e operações de álgebra linear, espaços amostrais e distribuições de probabilidade;
- **R**$(q_i$,$d_j)$ é uma função de similaridade que associa um número real à representação de uma consulta $q_i \in$ **Q** e de um documento $d_j \in$ **D**, definindo uma ordenação entre os documentos pertencentes a **D** em relação à consulta q_i.

No processo de construção de um modelo, a primeira coisa a ser feita é gerar as visões lógicas dos documentos e das consultas. A representação dos documentos pode ser dada como um subconjunto de todos os termos presentes neles. Algumas melhorias que normalmente são feitas nessa representação são a remoção de $stopwords$[1], a implementação de $stemming$[2] e a aplicação de filtros no vocabulário de acordo com o domínio do problema. A representação das consultas, por sua vez, passa pelo mesmo processamento dos documentos, podendo, adicionalmente, expandir seus termos.

A Figura 2.1 representa a quádrupla que constitui um modelo de RI. Os conjuntos de documentos e consultas são transformados em visões lógicas **D** e **Q**, que são modeladas utilizando um arcabouço teórico $\mathcal{F}$, a fim de gerar função de similaridade do modelo **R**. A função **R**$(q_i$, $d_j)$ calcula a similaridade de cada documento $d_j \in D$ em relação a uma dada consulta $q_i \in Q$, gerando uma pontuação numérica. Estas pontuações geradas são responsáveis pela ordenação final do *ranking* gerado pelo sistema de RI.

[1] Segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), *stopwords* consistem em palavras que não são úteis para o cálculo da relevância de um documento, tais como artigos e preposições.

[2] Segundo (MORAL et al., 2014), $stemming$ é o processo de diminuir o dicionário de uma coleção por meio da redução de palavras pela sua raiz gramatical.

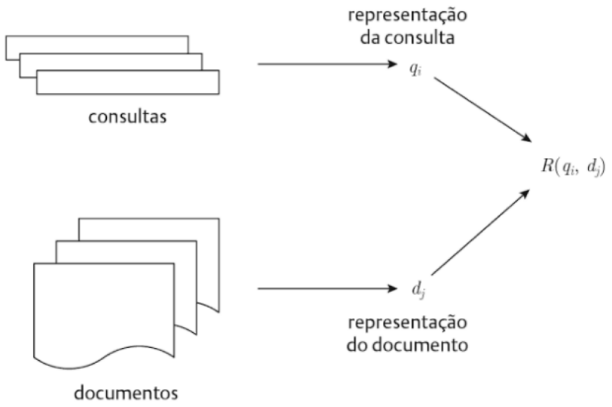

Figura 2.1 – Concepção da função de similaridade $\mathbf{R}(q_i, d_j)$ (BAEZA-YATES; RIBEIRO-NETO, 2013), p.23

Neste trabalho, adicionalmente, o conceito de **função de ranqueamento** foi definido. Diferentemente da função de similaridade, exibida na Figura 2.1, uma função de ranqueamento pode ser definida por meio da combinação de uma ou mais funções de similaridade de distintos modelos de RI. A combinação destas funções pode ser feita de forma automática por meio de, por exemplo, métodos de agregação de ranqueamento.

Modelos de RI são baseados, normalmente, em textos, pois eles usam o texto dos documentos para ranqueá-los em relação à consulta (JANEIRO, 2017). Porém, na Web, pode ser necessário também utilizar informações sobre estruturas de *links* para alcançar um bom nível de ranqueamento. Além disso, objetos multimídia não são codificados da mesma forma que textos. Por essas particularidades, textos, *links* e multimídia possuem suas próprias categorias de modelos de RI que são as baseadas em texto, as baseadas em *links* e as baseadas em objetos multimídia. A Figura 2.2 ilustra essas categorias e os modelos de RI mais reconhecidos em cada uma.

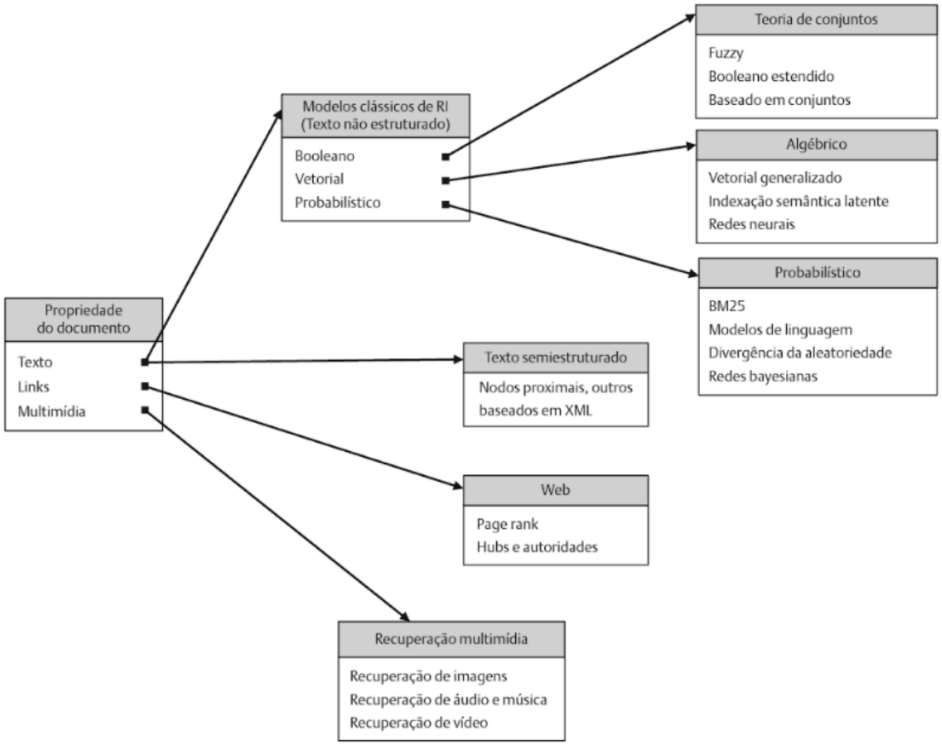

Figura 2.2 – Taxonomia dos modelos de RI (BAEZA-YATES; RIBEIRO-NETO, 2013), p.24

Os modelos de RI cuja categoria é baseada em texto, ao qual inclusive aplica-se ao objetivo deste trabalho, podem ser divididos entre modelos para textos não estruturados e textos semi-estruturados. Na categoria de textos não estruturados, o texto é modelado apenas como uma sequência de palavras; para esta categoria, modelos de RI denominados clássicos e estendidos foram propostos. Já na segunda categoria, os componentes da estrutura do texto (título, seções, capítulos, parágrafos, *links*) fazem parte integral do documento, e os modelos de RI propostos fazem uso da estruturação do texto para definição de sua função de similaridade.

2.1.2 Modelos Clássicos

Os modelos clássicos de RI apresentam conceitos distintos para construção da função de similaridade. Para isso, estes modelos consideram que cada documento é representando por um conjunto de termos de indexação. Ademais, a cada termo de indexação k_i presente no documento $d_j \in D$, é atribuída uma ponderação $w_{i,j}$ que quantifica a correlação do termo k_i com documento d_j. De maneira análoga, uma ponderação $w_{i,q}$ é atribuída aos termos de indexação k_i presentes na consulta $q \in Q$.

O restante desta Subseção apresenta a fundamentação matemática de cada modelo clássico de RI. Portanto, as Subseções 2.1.2.1, 2.1.2.2, 2.1.2.3 descrevem, respectivamente, o funcionamento dos modelos clássicos Booleano, Vetorial e Probabilístico.

2.1.2.1 Modelo Booleano

O Modelo Booleano (BM) é considerado um dos mais primitivos modelos para recuperação textual. Fundamentando-se na teoria dos conjuntos (GUDIVADA et al., 1997), o modelo é considerado altamente intuitivo e com semântica precisa. No modelo, a visão lógica dos documentos é representada por meio de um conjunto dos termos de indexação de tais documentos, enquanto as consultas são representadas como expressões lógicas. Seu adjetivo "booleano", portanto, refere-se ao uso da álgebra booleana, onde as palavras-chave de cada consulta podem ser combinadas por meio dos operadores lógicos *AND*, *OR*, e *NOT* (LASHKARI; MAHDAVI; GHOMI, 2009).

Os documentos recuperados pelo BM são aqueles que satisfazem a expressão lógica da consulta. Logo, não há satisfação parcial nos resultados de uma consulta, pois um documento só vai ser relevante se satisfazer completamente a expressão booleana. Desta forma, a similaridade de um documento pode ser apenas 1 (relevante) ou 0 (não relevante).

Ademais, segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), o modelo utiliza-se da ideia de correspondência exata entre os termos da consulta do usuário e os documentos da coleção, o que implica, na prática, que não há uma diferença de importância entre os termos de indexação. Desta forma, para o BM, a ponderação $w_{i,j}$, associada ao termo k_i e o documento d_j, é não negativa e binária. Todos os termos de indexação são considerados mutuamente independentes e, por isso, a ponderação $w_{i,j}$ pode ser formalizada da seguinte maneira:

- $w_{i,j}$ = 1, caso o termo k_i da consulta estiver presente no documento d_j;
- $w_{i,j}$ = 0, caso o termo k_i da consulta não estiver presente no documento d_j.

Desta forma, é possível representar os documentos e as consultas como componentes conjuntivos de termos, usando a ponderação $w_{i,j}$ associada a cada um deles. Considere $V = k_1, k_2, ..., k_t$ como o vocabulário de termos de toda a coleção. Se três termos de indexação k_l , k_m e k_n ocorrem em um mesmo documento d_j , diz-se que o padrão $[k_l, k_m, k_n]$ de coocorrência de termos foi observado em tal documento. Cada um destes padrões de coocorrência de termos é chamado de componente conjuntivo de termo. Para ilustrar, o componente conjuntivo de termo $(1, 0, ..., 0)$ indica, apenas, a presença do termo k_1.

Assim, a similaridade entre a consulta $q \in Q$ e o documento $d_j \in D$, para o BM, pode ser generalizada como:

$$sim(d_j, q) = \begin{cases} 1 & \text{se } \exists c(q) \mid c(q) = c(d_j) \\ 0 & \text{caso contrário} \end{cases} \tag{2.1}$$

onde $c(q)$ corresponde a qualquer componente conjuntivo da consulta q e $c(d_j)$ corresponde ao componente conjuntivo do documento d_j (BAEZA-YATES; RIBEIRO-NETO, 2013).

De acordo com (SOUZA, 2006), a principal desvantagem do modelo é o fato do mesmo trabalhar de forma binária. Com isso, na maioria das vezes, os resultados da busca podem ser um conjunto muito grande ou nulo de documentos. Além disso, segundo (KHAN, 2014), nem sempre é fácil, para a maioria dos usuários, transformar suas necessidades de informação em expressões booleanas, o que diminui drasticamente a usabilidade do modelo. Apesar disso, a simplicidade do modelo o torna uma solução computacionalmente eficiente.

2.1.2.2 Modelo Vetorial

O clássico Modelo Vetorial (VSM) reconhece as limitações do modelo BM e propõe uma solução algébrica que torna possível a realização de casamentos parciais nas consultas. Neste modelo, os termos de indexação são mutuamente independentes e são representados por meio de vetores em um espaço *t*-dimensional, onde *t* é o número de termos de indexação. A representação do documento e da consulta são, desta forma, vetores *t*-dimensionais dados por:

- $\overrightarrow{d_j} = (w_{1,j}, w_{2,j}, ..., w_{t,j})$
- $\overrightarrow{q} = (w_{1,q}, w_{2,q}, ..., w_{t,q})$

onde $w_{i,q}$ é a ponderação associada ao par termo-consulta (k_i, q), com $w_{i,q} \geq 0$ e $w_{i,j}$ corresponde à ponderação associada ao termo k_i e o documento d_j.

A representação vetorial dos documentos e consultas faz com que cada termo de indexação k_i seja representado como um componente do vetor por meio de uma ponderação, numérica não negativa e não binária (BAEZA-YATES; RIBEIRO-NETO, 2013). Esta ponderação corresponde a um número real que indica o quão relevante um termo é em relação ao documento ou à consulta (BERRY; DRMAC; JESSUP, 1999). Desta forma, para cada par termo-consulta (k_i, q), é atribuído um peso $w_{i,q}$, que compõe a representação vetorial da consulta $\overrightarrow{q}$. Da mesma forma, ao par termo-documento (k_i, d_j), é atribuído um peso $w_{i,j}$, que compõe a representação vetorial do documento $\overrightarrow{d_j}$.

A forma mais comum de se realizar o cálculo do grau de similaridade entre dois vetores é determinando sua proximidade por meio da medida do cosseno do ângulo entre eles (vide Figura 2.3).

A Figura 2.3 ilustra, de forma simplificada, a representação vetorial de um documento e de uma consulta, considerando um vocabulário com apenas 2 termos de indexação. Neste caso, o ângulo θ entre estes os vetores $\overrightarrow{d_j}$ e $\overrightarrow{q}$ significa, para o VSM, o grau de similaridade entre eles.

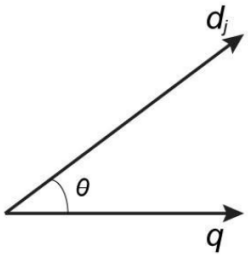

Figura 2.3 – Ângulo entre o vetor de um documento d_j e uma consulta q, no VSM (BAEZA-YATES; RIBEIRO-NETO, 2013), p.46

Desta forma, é possível generalizar a função de similaridade do modelo como:

$$sim(d_j, q) = \frac{\overrightarrow{d_j} \bullet \overrightarrow{q}}{|\overrightarrow{d_j}| \times |\overrightarrow{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}} \tag{2.2}$$

onde $|\overrightarrow{d_j}|$ e $|\overrightarrow{q}|$ são normas dos vetores do documento e da consulta e $\overrightarrow{d_j} \bullet \overrightarrow{q}$ é o produto interno dos vetores.

Analisando a Equação 2.2, pode-se notar que o fator $|\overrightarrow{q}|$ não afeta o ranqueamento, porque ele é o mesmo para todos os documentos. No entanto, o fator $|\overrightarrow{d_j}|$ faz a normalização pelo tamanho do documento, o que impede que documentos com uma grande quantidade de termos apareçam mais bem ranqueados do que documentos considerados menores. Existem formas de normalização mais sofisticadas, tais como a normalização pivotada (SINGHAL; BUCKLEY; MITRA, 2017).

Diferentemente do BM, o VSM atribui uma ponderação não binária aos termos de índice da consulta e do documento. Desta forma, o modelo afirma que nem todos termos são igualmente úteis na descrição do conteúdo de um documento, ou seja, o modelo torna possível que termos diferentes tenham ponderações diferentes. Um termo, por exemplo, que está presente em todos os documentos de uma coleção, pouco acrescenta em uma consulta; no entanto, um termo que aparece em poucos documentos reduz o conjunto que possa ser de interesse do usuário e deve ser valorizado pela consulta. Tais ponderações, no VSM, são calculados por meio do esquema denominado **TF-IDF** (JONES, 1972; SALTON; YANG, 1973).

De acordo com a Hipótese de Luhn[3], os termos com alta frequência no documento são importantes para descrever os tópicos-chave de um documento. A ponderação **TF** (*term frequency*) baseia-se diretamente nesta hipótese, a qual leva à seguinte formulação:

$$tf_{i,j} = f_{i,j} \tag{2.3}$$

[3] De acordo com (BAEZA-YATES; RIBEIRO-NETO, 2013), o peso de um termo k_i que ocorre em um documento d_j é diretamente proporcional à frequência do termo $f_{i,j}$.

onde $f_{i,j}$ é a frequência do termo k_i no documento d_j. Esta ponderação possui distintas variantes (vide Tabela 2.1) que utilizam, basicamente, a frequência $f_{i,j}$ para realizar o cálculo de sua ponderação.

Tabela 2.1 – Variantes da ponderação **TF** (BAEZA-YATES; RIBEIRO-NETO, 2013), p.41

Esquema de Ponderação	Peso TF
binário	$\{0, 1\}$
frequência bruta	$f_{i,j}$
normalização logarítmica	$1 + \log f_{i,j}$
normalização dupla 0.5	$0.5 + 0.5\frac{f_{i,j}}{max_i f_i i,j}$

A Tabela 2.1 mostra algumas diferentes formas de se realizar o cálculo da ponderação **TF** para os termos. A variante binária comporta-se da mesma forma que a ponderação utilizada no BM: o **TF** vale 1 se o termo ocorre no documento, e 0, caso contrário. A variante da frequência bruta constitui a concepção base da ponderação **TF**, baseada na Hipótese de Luhn, isto é, na contagem direta de frequência. A normalização logarítmica, por sua vez, utiliza logaritmos para atribuir incrementos que diminuem à medida que a frequência aumenta. A variante normalização dupla 0,5 normaliza os pesos pela frequência máxima em um documento $max_i f_i i, j$ e, também, normaliza o peso para que ele fique entre 0,5 e 1.

A medida **TF** está diretamente relacionada com a especificidade[4] dos termos de índice. No entanto, o **TF** não faz distinção entre termos que ocorrem em todos o documentos da coleção e termos que ocorrem somente em alguns documentos. Sabe-se intuitivamente que um termo que aparece em toda coleção terá, provavelmente, pouca utilidade para identificar a relevância dos documentos. Portanto, para um cálculo preciso do peso de um determinado termo de indexação, é necessária uma estatística global que caracterize o termo em relação a toda coleção (FERNEDA, 2003). Esta ponderação é chamada de **IDF** (*inverse document frequency*) e pode ser calculada, comumente, para um determinado termo k_i da coleção da seguinte forma:

$$idf_i = \frac{N}{n_i} \tag{2.4}$$

onde N corresponde ao número de documentos da coleção e n_i corresponde ao número de documentos associados ao termo k_i.

A ponderação **IDF** de um termo pode ser calculada de outras formas (vide Tabela 2.2) que, basicamente, comparam o número de documentos em que um termo de índice está associado. Desta forma, quanto menor o número de documentos que contêm um determinado termo, maior o **IDF** desse termo. Se todos os documentos da coleção contiverem um determinado termo, o **IDF** desse termo será igual a um (1), correspondendo ao menor valor possível.

[4] Segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), especificidade é uma propriedade dos termos de índice que pode ser interpretada como o quão bem determinado termo descreve o tópico de um documento.

Tabela 2.2 – Variantes da ponderação **IDF** (BAEZA-YATES; RIBEIRO-NETO, 2013), p.41

Esquema de Ponderação	Peso IDF
Unário	1
Frequência Inversa	$\log \frac{N}{n_i}$
Frequência Inversa Suave	$\log(1 + \frac{N}{n_i})$
Frequência Inversa Máxima	$\log(1 + \frac{max_i n_i}{n_i})$
Frequência Inversa Probabilística	$\log(\frac{N-n_i}{n_i})$

Para os pesos **IDF**, cinco variantes são ilustradas na Tabela 2.2. A variante unária atribui 1 ao **IDF** de todos os termos. A variante de frequência inversa é a formulação padrão já discutida. A variante frequência inversa suave, por sua vez, soma 1 à fração, a fim de evitar comportamentos estranhos com valores extremos de n_i. Já a variante da frequência inversa máxima computa o peso relativo ao termo com maior frequência de documento, em vez de usar o número de documentos na coleção. Por fim, a variante frequência inversa probabilística é derivada da teoria de probabilidades.

Finalmente, o peso $w_{i,j}$ de um termo k_i em relação a um documento d_j, utilizado no VSM, pode ser definido como:

$$w_{i,j} = tf_{i,j} \times idf_i \tag{2.5}$$

caracterizando, assim, o esquema de ponderação **TF-IDF**. Para as consultas, o cálculo é feito de forma análoga:

$$w_{i,q} = tf_{i,q} \times idf_i \tag{2.6}$$

Vale ressaltar que, não necessariamente, o mesmo esquema de ponderação **TF-IDF** é utilizado para modelar os documentos e as consultas. No entanto, neste trabalho, o ambiente proposto utiliza o mesmo esquema de ponderação para documentos e consultas como base para execução dos experimentos.

Por fim, temos que o VSM é considerado uma boa estratégia para coleções genéricas. Utilizando esquemas de ponderação, além da recuperação parcial de documentos, o modelo apresenta resultados que dificilmente podem ser melhorados sem o uso de técnicas como expansão de consultas ou realimentação de relevância (BAEZA-YATES; RIBEIRO-NETO, 2013).

2.1.2.3 Modelo Probabilístico

O último modelo clássico, proposto por (ROBERTSON; JONES, 1976), é o Probabilístico (PM). Sua ideia fundamental, segundo (FERNEDA, 2003), consiste em, dada uma consulta, dividir a coleção (com N documentos) em quatro subconjuntos distintos (vide Figura 2.4): o conjunto dos documentos relevantes (R), o conjunto dos documentos recuperados (A), o conjunto

dos documentos relevantes que foram recuperados ($A \cap R$) e o conjunto dos documentos não relevantes e não recuperados ($N - (A \cup R)$). Por fim, tem-se que N, por simplicidade, corresponde ao conjunto de todos documentos da coleção (vide Figura 2.4) e $\overline{R}$ constitui o conjunto dos documentos não relevantes.

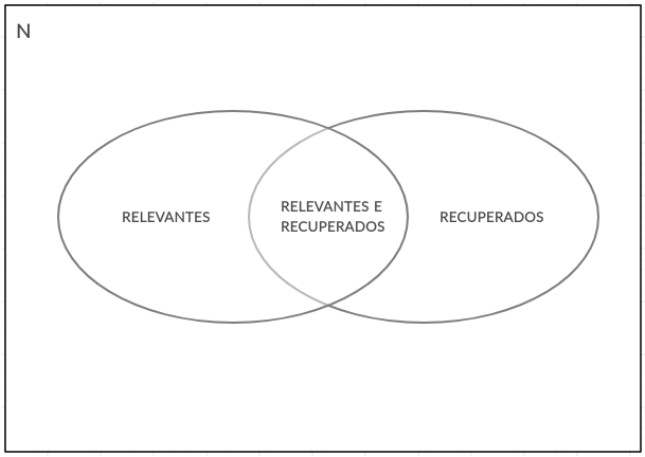

Figura 2.4 – Subconjuntos de documentos após a execução de uma busca. Fonte: do próprio autor

Para o PM, o resultado ideal de uma busca q é o conjunto que contenha todos e apenas os documentos relevantes para o usuário, isto é, todo o conjunto R. Assim o processo de consulta pode ser interpretado como um processo de especificação das propriedades deste conjunto ideal R. O maior problema é que tudo o que se sabe, em um primeiro momento, é que existem termos de indexação os quais as semânticas devem ser utilizadas para especificação dessas propriedades. Desta forma, o modelo tenta adivinhar tais características por meio da formulação de uma expressão de busca, gerando uma primeira descrição probabilística desse conjunto.

Com a finalidade de melhorar a descrição probabilística do conjunto de resposta ideal, uma interação com o usuário é iniciada, após a primeira tentativa do modelo. O usuário pode, por exemplo, visualizar os documentos recuperados na primeira iteração e decidir quais são relevantes e quais não. O modelo pode, então, utilizar dessas informações para refinar a descrição do conjunto de resposta ideal. Repetindo este processo muitas vezes, espera-se que, gradativamente, a descrição do conjunto ideal fique mais precisa e, por consequência, os resultados da busca também.

Segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), dada uma consulta q, o PM atribui a cada documento d_j, como medida de similaridade com a consulta, a razão P(d_j relevante a q) / P(d_j não relevante a q) que computa a chance do documento d_j ser relevante à consulta q. Tomando-se a chance de relevância como pontuação para o ranqueamento, a probabilidade de um julgamento errado é minimizada.

Desta forma, no PM, uma consulta q é um subconjunto dos termos de indexação. Um documento d_j pode ser representado por um vetor de pesos binários que indicam a presença ou

ausência de termos de indexação, como segue:

$$\overrightarrow{d_j} = (w_{1,j}, w_{2,j}, ..., w_{t,j}) \tag{2.7}$$

onde $w_{i,j} = 1$ se o termo k_i ocorre no documento d_j e $w_{i,j} = 0$ caso contrário. Assim, define-se $P(\overrightarrow{d_j}|R, q)$ como a probabilidade de que o documento d_j seja relevante para a consulta q. Além disso, $P(\overrightarrow{d_j}|\overline{R}, q)$ é a probabilidade de que o documento d_j não seja relevante para a consulta q. A similaridade $sim(d_j, q)$ entre o documento d_j e a consulta q pode ser dada por:

$$sim(d_j, q) = \frac{P(\overrightarrow{d_j}|R, q)}{P(\overrightarrow{d_j}|\overline{R}, q)} \tag{2.8}$$

A partir da Equação 2.8, define-se $P(k_i|R, q)$ como probabilidade do termo de índice k_i estar presente em um documento selecionado de forma aleatória do conjunto R, e $P(\overline{k_i}|R, q)$ a probabilidade do termo de índice k_i não estar presente em tal conjunto. As probabilidades associadas ao conjunto $\overline{R}$ têm significados análogos. Para simplificar, define-se:

$$p_{iR} = P(k_i|R, q)$$
$$q_{iR} = P(k_i|\overline{R}, q)$$

Assumindo-se que os termos de indexação são independentes e utilizando o teorema de Bayes[5], obtém-se, a partir da Equação 2.8:

$$sim(d_j, q) \sim \sum_{k_i \in q \wedge k_i \in d_j} log\left(\frac{p_{iR}}{1 - p_{iR}}\right) + log\left(\frac{1 - q_{iR}}{q_{iR}}\right) \tag{2.9}$$

Atualmente, existem diversas abordagens para definir, inicialmente, as probabilidades p_{iR} e q_{iR}. (CROFT; HARPER, 1979) propuseram um método de *ranqueamento na ausência de informação de relevância*, que considera duas suposições iniciais:

1. p_{iR} é constante igual a 0,5 indicando nenhum conhecimento prévio quanto à sua distribuição;

2. a distribuição dos termos de indexação entre os documentos não relevantes pode ser aproximada pela distribuição dos termos de indexação entre todos os documentos da coleção.

[5] Segundo (SPIEGEL; SCHILLER; SRINIVASAN, 2016), o teorema de Bayes diz que se $\{A_1, A_2, ..., A_k\}$ são eventos mutuamente exclusivos, cuja união é o espaço amostral S, tem-se a seguinte formulação

$$P(A_k|A) = \frac{P(A_k)P(A|A_k)}{\sum_{j=1}^{n} P(A_j)P(A|A_j)}$$

Estas suposições geram

$$p_{iR} = 0.5 \quad q_{iR} = \frac{n_i}{N}$$

Substituindo na Equação 10, tem-se:

$$sim(d_j, q) \sum_{k_i \in q \wedge k_i \in d_j} log\left(\frac{N - n_i}{n_i}\right) \tag{2.10}$$

A partir desta estimativa inicial, é possível recuperar os documentos e fornecer uma ordenação probabilística para eles. Para melhorar o ranqueamento gerado, (CROFT; HARPER, 1979) propuseram, adicionalmente, um algoritmo iterativo para estimar novos valores p_{iR} e q_{iR}.

Por fim, temos que o PM possui como maior vantagem sua otimalidade, ou seja, os documentos são ranqueados de forma decrescente de acordo com sua probabilidade de serem relevantes, com base na informação disponível ao sistema. No entanto, o modelo se baseia-se em suposições que, segundo (PANNU; JAMES; BIRD, 2014), nem sempre representarão bem a realidade.

2.1.3 Modelos Estendidos

Baseando-se nos modelos clássicos, outros modelos, denominados estendidos, foram propostos. Estes modelos utilizam-se dos conceitos clássicos de recuperação textual e, baseando neles, criam novas soluções para o cálculo de similaridade entre documentos e consultas. Aqui, são tratados, de forma breve, os modelos Booleano Estendido (vide Subseção 2.1.3.1), Vetorial Generalizado (vide Subseção 2.1.3.2), BM25 (vide Subseção 2.1.3.3), DFree (vide Subseção 2.1.3.4), PL2 (vide Subseção 2.1.3.4), Redes de Crença (vide Subseção 2.1.3.5) e PageRank (vide Subseção 2.1.3.6).

2.1.3.1 Modelo Booleano Estendido

O modelo Booleano Estendido (EBM), proposto por (SALTON; FOX; WU, 1983), é um modelo híbrido que tenta unir a potencialidade das expressões booleanas com a precisão do VSM. Por um lado, busca-se flexibilizar o BM, introduzindo o casamento parcial no cálculo de similaridade e, por outro lado, busca-se dar maior poder às buscas do VSM, por meio do uso dos operadores booleanos (FERNEDA, 2003).

O julgamento binário, inerente ao BM, não está de acordo com o senso comum. Intuitivamente sabe-se que, após uma busca utilizando uma expressão booleana conjuntiva ($k_1 \wedge k_2$), os documentos que possuem apenas um dos termos da expressão, que não foram recuperados, possuem um certo grau de importância e poderiam vir a ser considerados relevantes por um usuário. De forma similar, utilizando uma expressão disjuntiva ($k_1 \vee k_2$), um documento que

possui ambos os termos da expressão pode ser considerado mais importante do que os documentos que possuem apenas um termo.

Segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), quando apenas dois termos são considerados, é possível plotar os documentos e consultas em um mapa bidimensional (vide Figura 2.5). Um documento d_j é posicionado nesse espaço por meio da adoção de pesos $w_{x,j}$ e $w_{y,j}$ associados aos pares (k_x , d_j) e (k_y , d_j), respectivamente. Considere os pesos $w_{x,j}$ e $w_{y,j}$, obtidos por meio de qualquer esquema de ponderação **TF-IDF** (vide Subseção 2.1.2.2), e que estejam normalizados, isto é, assumam valores entre 0 e 1.

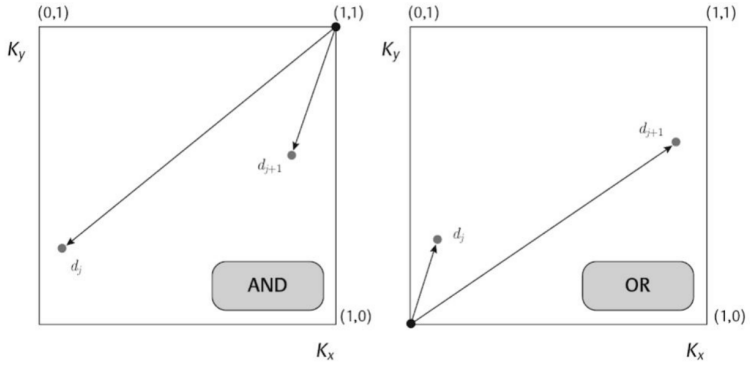

Figura 2.5 – Lógica Booliana Estendida considerando o espaço composto por apenas dois termos k_x e k_y (BAEZA-YATES; RIBEIRO-NETO, 2013), p.63

Observando a Figura 2.5, nota-se que, em expressões disjuntivas, o ponto (0, 0) deve ser evitado, pois representa a situação na qual nenhum dos termos está presente no documento. Assim, a distância de um documento ao ponto (0,0) pode ser considerada como grau de similaridade do documento em relação à busca. Já para expressões conjuntivas, o ponto (1,1) é o mais desejável, já que representa a situação na qual ambos os termos da expressão estão presentes na representação de um documento. Deste modo, o complemento da distância ao ponto (1,1) pode ser tomado como medida de similaridade.

Assim, essas distâncias podem ser normalizadas, gerando:

$$sim(q_{or}, d) = \sqrt{\frac{x^2 + y^2}{2}} \quad sim(q_{and}, d) = 1 - \sqrt{\frac{(1-x)^2 + (1-y)^2}{2}} \tag{2.11}$$

Este modelo pode ser naturalmente estendido, para considerar distâncias euclidianas em um espaço com t dimensões, onde t é o número de termos de índice para uma coleção qualquer. Contudo, uma generalização mais compreensiva seria adotar a teoria das normas dos vetores (BAEZA-YATES; RIBEIRO-NETO, 2013).

Visando tornar o modelo mais flexível, o conceito matemático de *p-norm* é utilizado. Este conceito generaliza a noção de distância para incluir não apenas as distâncias euclidianas, mas também distâncias p, onde $1 \leq p \leq \infty$ é um novo parâmetro cujo valor precisa ser especificado

em tempo de consulta. Por exemplo, dado um vetor $\overrightarrow{v} = (v_1, v_2, ..., v_n)$, sua *p-norm* pode ser calculada como:

$$|\overrightarrow{v}|_p = (v_1^p + v_2^p + ... + v_n^p)^{\frac{1}{p}}$$

Desta forma, a similaridade entre um documento e uma consulta continua sendo uma função da distância entre dois pontos. Porém, ao invés de ser utilizar a distância euclidiana, utiliza-se a *p-norm*. Assim, considerando $w_{i,j}$ como o peso associado ao par (k_i , d_j), as similaridades entre consulta e documentos podem ser dadas, de forma generalizada, por:

$$sim(d_j, q_{or}) = \left(\frac{x_1^p + x_2^p + ... + x_m^p}{m}\right)^p \tag{2.12}$$

$$sim(d_j, q_{and}) = 1 - \left(\frac{(1 - x_1)^p + (1 - x_2)^p + ... + (1 - x_m)^p}{m}\right)^{\frac{1}{p}} \tag{2.13}$$

Segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), esse procedimento pode ser aplicado recursivamente, independente do número de operadores AND / OR. Porém, se a operação de disjunção sobre a operação de conjunção for distribuída e as pontuações forem recalculadas, serão encontrados valores diferentes. Isto é, os operadores lógicos booleanos não preservam o ranqueamento, o que é uma desvantagem do EBM.

A *p-norm* possui algumas particularidades que valem apena serem ressaltadas. Primeiro, para $p = 1$, as consultas conjuntivas e disjuntivas são avaliadas pela soma de pesos dos termos nos documentos assim como é feito no VSM. Além disso, para $p = \infty$, as consultas são avaliadas de forma parecida com o BM. Dessa forma, pode-se variar o comportamento do ranqueamento de forma que ele se aproxime mais do VSM ou do BM.

Por fim, conclui-se que o EBM tenta contornar as limitações do VSM e do BM clássico por meio de uma conceituação matemática mais genérica. As expressões booleanas e as buscas vetoriais são casos particulares do EBM. No entanto, a adição de um novo parâmetro p pode tornar o modelo mais complexo para ser compreendido e executado.

2.1.3.2 Modelo Vetorial Generalizado

Conforme já discutido, os três modelos clássicos de RI supõem a independência entre os termos de indexação. Para o VSM, por exemplo, essa suposição implica que os vetores unitários do conjunto $\{\overrightarrow{k_1}, \overrightarrow{k_2}, ..., \overrightarrow{k_t}\}$ são linearmente independentes, significando a ortogonalidade par a par dos termos de índice (BAEZA-YATES; RIBEIRO-NETO, 2013). Deste modo, (WONG; ZIARKO; WONG, 1985) propuseram uma interpretação alternativa do VSM, no qual os vetores dos termos de indexação não são ortogonais par a par. Em vez disso, eles são formados por componentes *menores* derivados da coleção. Tal interpretação levou ao surgimento do modelo estendido Vetorial Generalizado (GVSM).

Considere um vocabulário $V = \{k_1, k_2, ..., k_t\}$ de termos de índice na coleção. A partir dele, é possível gerar 2^t componentes conjuntivos de termo, tal como no BM clássico. Cada componente conjuntivo é chamado, no GVSM, de *minterm* m_r. A Tabela 2.3 mostra a lista dos *minterms* para o vocabulário V.

Tabela 2.3 – *Minterms* definidos para um vocabulário V de tamanho t

	$(k_1, k_2, k_3, ..., k_t)$
m_1	(0, 0, 0, ..., 0)
m_2	(1, 0, 0, ..., 0)
.	
.	
.	
m_{2^t}	(1, 1, 1, ..., 1)

Note que, por meio da Tabela 2.3, tem-se que o *minterm* m_1 indica um padrão de co-ocorrência entre os termos comuns aos documentos que não contém termos de indexação. O *minterm* m_2 indica um padrão de coocorrência entre termos referente aos documentos que contém apenas k_1. Por fim, o *minterm* m_{2^t} indica documentos que possuem todos termos de índice (BAEZA-YATES; RIBEIRO-NETO, 2013).

Define-se, por meio do conceito de *minterms*, a função:

$$on(i, m_r) = \begin{cases} 1 & \text{se } k_i \text{ está incluído em } m_r \\ 0 & \text{caso contrário} \end{cases}$$

onde se nota que $on(1, m_2) = 1$ e $on(i, m_2) = 0$ para todos $i \geq 2$. Com isso, pode-se afirmar que, segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), para qualquer documento d_j existe um *minterm* que inclui exatamente os termos que ocorrem no documento e nenhum outro. Este *minterm* é denominado $c(d_j)$.

Por meio deste novo conceito, pode-se associar cada *minterm* m_r a um vetor unitário $\overrightarrow{m_r}$ que será utilizado como base ortogonal para o GVSM (vide Tabela 2.4).

Tabela 2.4 – Conjunto de vetores unitários *minterms* $\overrightarrow{m_r}$.

	1, 2, 3, ..., 2^t
$\overrightarrow{m_1}$	(1, 0, 0, ..., 0)
$\overrightarrow{m_2}$	(0, 1, 0, ..., 0)
.	
.	
.	
$\overrightarrow{m_{2^t}}$	(0, 0, 0, ..., 1)

Assim, com base na Tabela 2.4, os conjuntos de vetores $\overrightarrow{m_r}$ são, por definição, ortogonais par a par. Esta ortogonalidade, por sua vez, não significa a independência entre os termos

de índice. Ao contrário, os termos de indexação estão agora correlacionados por vetores $\overrightarrow{m_r}$ (BAEZA-YATES; RIBEIRO-NETO, 2013).

Desta forma, para determinar o vetor $\overrightarrow{k_i}$ associado ao termo de indexação k_i, basta simplesmente somar os vetores para todos *minterms* m_r que incluem o termo k_i e normalizar:

$$\begin{aligned} \overrightarrow{k_i} &= \frac{\sum_{\forall r} on(i, m_r) \times c_{i,r} \times \overrightarrow{m}_r}{\sqrt{\sum_{\forall r} on(i, m_r) \times c_{i,r}^2}} \\ c_{i,r} &= \sum_{d_j | c(d_j) = m_r} w_{i,j} \end{aligned} \tag{2.14}$$

onde $w_{i,j}$ é obtido por meio dos esquemas de ponderação **TF-IDF** já mencionados. Nota-se que, para o GVSM não ficar muito custoso, não é necessário realizar os cálculos considerando todos os *minterms* possíveis; basta considerar, apenas, os *minterms* ativos, isto é, aqueles *minterms* que possuem pelo menos um documento na coleção que possui o seu padrão de ocorrência de termos. Assim, a computação do *ranking* não depende mais de um número exponencial de *minterms*.

Com isso, o GVSM torna-se capaz de representar a relação entre dois termos de indexação k_i e k_j, por meio do produto interno entre os termos $\overrightarrow{k_i} \bullet \overrightarrow{k_j}$ (BAEZA-YATES; RIBEIRO-NETO, 2013). Por fim, aplicando os conceitos do VSM e a representação dos termos de indexação apresentada na Equação 2.14, tem-se que a representação dos documentos e das consultas pode ser dada por:

$$\overrightarrow{d_j} = \sum_{\forall i} w_{i,j} \times \overrightarrow{k_i} \quad \text{e} \quad \overrightarrow{q} = \sum_{\forall i} w_{i,q} \times \overrightarrow{k_i}$$

Além disso, o cálculo de similaridade entre um documento d_j e uma consulta q, no GVSM, é dado por:

$$sim(d_j, q) = \frac{\overrightarrow{d_j} \bullet \overrightarrow{q}}{|\overrightarrow{d_j}| \times |\overrightarrow{q}|} \tag{2.15}$$

onde $sim(d_j, q)$ corresponde à mesma equação utilizada pelo VSM, isto é, a medida do cosseno do ângulo formado pelos vetores do documento e da consulta em questão.

O modelo GVSM combina as boas e precisas características do modelo clássico VSM a um esquema que garante a correlação entre os termos de indexação. Contudo, não se sabe ao certo se a correlação entre os termos traz melhorias na qualidade da recuperação, ou seja, não está claro na literatura em quais situações o modelo GVSM supera o VSM. Outro aspecto importante a se considerar é o custo computacional envolvido para o cálculo das correlações utilizando os *minterms*. Segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), para grandes coleções, o número

de *minterms* ativos pode ser diretamente proporcional ao número de documentos na coleção, o que torna o modelo GVSM muito mais custoso computacionalmente em relação aos demais modelos já mencionados.

2.1.3.3 Modelo BM25

O modelo BM25, proposto em (ROBERTSON et al., 1995; ROBERTSON; ZARAGOZA; TAYLOR, 2004), é um modelo probabilístico alternativo resultante de uma série de estudos feitos em cima da formulação probabilística clássica (vide Equação 2.10). Em linhas gerais, foi identificado que uma boa ponderação dos termos deve ser feita baseada em três princípios advindos do modelo VSM: frequência dos termos, frequência inversa dos termos e normalização pelo tamanho dos documentos. Note que a formulação clássica probabilística cobre apenas a frequência inversa dos termos.

Assim, a primeira ideia proposta para melhorar o ranqueamento probabilístico clássico foi a introdução de um fator de frequência dos termos $F_{i,j}$:

$$F_{i,j} = S_1 \times \frac{f_{i,j}}{K_1 + f_{i,j}} \tag{2.16}$$

onde $f_{i,j}$ é a frequência do termo k_i no documento d_j, K_1 é uma constante experimentalmente definida e S_1 é uma constante de escala relativa à K_1, normalmente dada como $S_1 = (K_1 + 1)$. Em seguida, foi introduzida a normalização pelo tamanho dos documentos, modificando a equação acima:

$$F'_{i,j} = S_1 \times \frac{f_{i,j}}{\frac{K_1 \times len(d_j)}{avg\_doclen} + f_{i,j}} \tag{2.17}$$

onde $len(d_j)$ é o tamanho do documento d_j (número de termos) e avg_doclen é a média do tamanho dos documentos na coleção. A introdução destes fatores levaram ao surgimento de diversas fórmulas BM (*Best Match*) tais como BM1, BM11 e, obviamente, BM25 (BAEZA-YATES; RIBEIRO-NETO, 2013). A variante BM25 define, a partir dos conceitos já apresentados, um novo fator $B_{i,j}$:

$$B_{i,j} = \frac{(K_1 + 1) f_{i,j}}{K_1 \left[(1 - b) + b \frac{len(d_j)}{avg\_doclen} \right] + f_{i,j}} \tag{2.18}$$

onde b é uma nova constante empírica introduzida com valores no intervalo entre 0 e 1. Desta forma, a equação de ranqueamento produzida pelo modelo BM25 pode ser escrita, tendo como base a formulação clássica do PM, como:

$$sim(d_j, q) \sim \sum_{k_i [q, d_j]} B_{i,j} \times log \left(\frac{N - n_i + 0.5}{n_i + 0.5} \right) \tag{2.19}$$

Ao contrário da concepção original do modelo PM, a fórmula do BM25 pode ser computada sem nenhuma informação de relevância fornecida pelo usuário, isto é, de forma completamente automática. Ademais, existe um consenso, segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), que um modelo BM25 devidamente ajustado fornece resultados melhores que o VSM para coleções genéricas. Assim, o BM25 tem sido amplamente usado como comparativo em novos modelos de ranqueamento na literatura de RI.

2.1.3.4 Modelos de Divergência da Aleatoriedade

Os modelos estendidos baseados em Divergência da Aleatoriedade (DFR) são modelos probabilísticos alternativos com características de modelos de linguagem, propostos por (AMATI; RIJSBERGEN, 2002). Sua ideia fundamental é derivar a ponderação dos termos medindo a divergência da distribuição de termos real em relação à distribuição de um termo produzida por meio de um processo aleatório.

Estes modelos baseiam-se em duas hipóteses principais. A primeira hipótese, proposta por (AMATI; RIJSBERGEN, 2002), é que nem todas palavras são igualmente importantes para descrever o conteúdo dos documentos. Por exemplo, palavras que estão distribuídas aleatoriamente por toda coleção carregam pouca informação. Dado um termo k_i, sua distribuição de probabilidade na coleção é dada por $P(k_i|D)$ e a quantidade de informação associada a ele é $-log(P(k_i|D))$. Diferentes especificações de distribuição de probabilidade geram diferentes modelos.

Além disso, a segunda hipótese proposta por (AMATI; RIJSBERGEN, 2002) é que uma distribuição complementar do termo pode ser obtida considerando apenas o subconjunto de documentos que contém o termo k_i. Esse subconjunto é chamado de conjunto elite. Desta forma, quanto menor a probabilidade de se observar um termo k_i em um documento d_j, isto é, quanto menor $P(k_i|d_j)$, mais raro e importante o termo é considerado no documento. Assim, a quantidade de informação do termo no conjunto elite é dada por $1 - P(k_i|d_j)$ (BAEZA-YATES; RIBEIRO-NETO, 2013).

Juntando essas hipóteses, o peso $w_{i,j}$ do termo k_i em um documento d_j é dado por:

$$w_{i,j} = (-log(P(k_i|D))) \times (1 - P(k_i|d_j)) \tag{2.20}$$

Assim, a equação de ranqueamento produzida por modelos DFR pode ser formulada como:

$$sim(d_j, q) = \sum_{k_i \in q} f_{i,q} \times w_{i,j} \tag{2.21}$$

onde $f_{i,q}$ é a frequência do termo k_i na consulta q.

Para computar a distribuição dos termos na coleção e a distribuição dos termos sobre o conjunto elite, diferentes modelos probabilísticos podem ser considerados, resultando em diferentes modelos de RI. Além disso, o arcabouço geral para definição de modelos DFR faz

necessária a aplicação de normalização da probabilidade gerada e, também, a aplicação de normalização sobre as frequências do termo (AMATI; RIJSBERGEN, 2002).

Desta forma, diferentes modelos DFR foram propostos na literatura. Neste trabalho, dois modelos utilizados na literatura e propostos pela ferramenta Terrier (OUNIS et al., 2005) foram considerados. São eles: PL2 e DFRee (DFR *free of parameters*).

O modelo PL2 possui probabilidades baseadas na distribuição de Poisson com efeito posterior de Laplace e aplicação de normalização (HE; OUNIS, 2005). Este modelo pode ser usado para tarefas que requerem boa precisão para os primeiros resultados de forma eficiente. Sua formulação pode ser dada por:

$$sim(d_j, q) = \sum_{k_i \in q} f_{i,q} \times \frac{1}{tfn+1}(tfn \times log_2 \frac{tfn}{\lambda} + (\lambda + \frac{1}{12 \times tfn} - tfn) \times log_2 e + 0.5 \times log_2(2\pi \times tfn)) \tag{2.22}$$

onde λ é uma constante relacionada à distribuição de Possion e *tfn* é a frequência do termo normalizada (HE; OUNIS, 2005). A normalização da frequência do termo para este modelo é dada por:

$$tfn = tf \times log_2(1 + c \times \frac{avg_l}{l}), \quad (c > 0) \tag{2.23}$$

onde l é o tamanho do documento, avg_l é tamanho médio dos documentos e c é uma constante experimentalmente definida.

Por fim, o modelo DFRee constitui uma solução alternativa proposta pela ferramenta Terrier de um modelo DFR onde não há a necessidade de especificar parâmetros de entrada[6].

Desta forma, estes modelos baseados em DFR constituem uma estrutura probabilística com o potencial de construir uma série de modelos eficazes para RI. Além disso, estes modelos, conforme analisado em (AMATI; RIJSBERGEN, 2002), são extremamente interessantes em relação a outras técnicas de RI para a tarefa de expansão de consultas.

2.1.3.5 Modelo de Rede de Crença

O modelo de Redes de Crença (BNM), descrito por (RIBEIRO; MUNTZ, 1996), é considerado um modelo probabilístico alternativo, baseado no modelo de Rede de Inferência (TURTLE; CROFT, 1989; TURTLE; CROFT, 1991).

De forma breve, pode-se definir o modelo de BNM como um modelo probabilístico que associa variáveis aleatórias aos termos de indexação, aos documentos e às consultas. Uma variável aleatória associada a um documento d_j representa o evento de observar aquele documento.

[6] Para mais informações: <http://terrier.org/>

Desta forma, a observação do documento d_j expressa uma crença sobre as variáveis aleatórias associadas aos seus termos de índice (BAEZA-YATES; RIBEIRO-NETO, 2013).

Diferentemente do modelo de Rede de Inferência, a topologia da rede no modelo BNM separa as porções de rede referentes aos documentos das porções referentes às consultas (vide Figura 2.6).

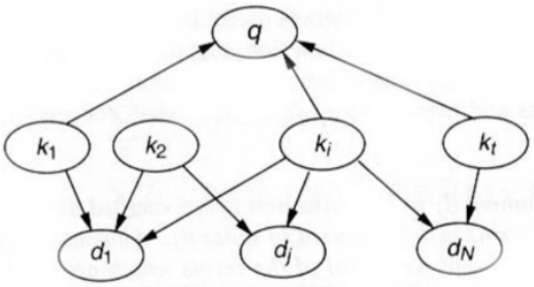

Figura 2.6 – Modelo básico de Rede de Crença (BAEZA-YATES; RIBEIRO-NETO, 2013), p.96

Analisando a Figura 2.6, tem-se que a consulta q é modelada como uma variável binária aleatória que é apontada pelos nodos dos termos de indexação que a compõe. Os documentos, por sua vez, são tratados de forma análoga.

Em uma rede de crença, todas as variáveis são binárias, o que parece arbitrário, mas, segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), isso simplifica a modelagem. Além disso, cada termo de indexação é visto como um *conceito elementar* e o conjunto de todos os termos de indexação é visto como um *espaço conceitual*. Qualquer subconjunto dos termos de indexação é interpretado como um conceito.

Assim, a pontuação de um documento d_j em relação a uma consulta q é interpretado como um relacionamento de correspondência entre conceitos, isto é, a classificação do documento d_j relativa à consulta q é interpretada como o quanto a consulta q cobre o documento d_j.

No modelo BNM, $P(d_j|q)$ é adotado como o grau de similaridade de um documento d_j em relação à consulta q. Aplicando as regras de Bayes e instanciando as variáveis aleatórias dos termos de índice, o que os torna mutuamente independentes, a probabilidade do documento d_j ser relevante para a consulta q é estabelecida por:

$$P(d_j|q) \sim \sum_{\forall \overrightarrow{k}} P(d_j|\overrightarrow{k}) \times P(q|\overrightarrow{k}) \times P(\overrightarrow{k}) \tag{2.24}$$

onde $\overrightarrow{k}$ é um vetor de t dimensões definido por $\overrightarrow{k} = (k_1, k_2, ..., k_t)$, e $k_1, k_2, ..., k_t$ são as variáveis aleatórias binárias da rede. Essas variáveis definem 2^t estados possíveis para $\overrightarrow{k}$.

Para que o modelo se torne aplicável, deve-se definir uma estratégia de classificação que, associada à rede de crenças, permita a recuperação ordenada dos documentos. Ou seja,

é necessário conceituar as probabilidades condicionais $P(d_j|\overrightarrow{k})$ e $P(q|\overrightarrow{k})$. Neste trabalho, a abordagem utilizada é baseada no modelo VSM. Para isso, um vetor $\overrightarrow{k_i}$ é definido por:

$$\overrightarrow{k_i} = \overrightarrow{k} \quad | \quad on(i, \overrightarrow{k}) = 1 \quad \wedge \quad \forall_{j \neq i} on(i, \overrightarrow{k}) = 0$$

onde $on(i, \overrightarrow{k})$ é dado por:

$$on(i, \overrightarrow{k}) \begin{cases} 1 & \text{se } k_i = 1 \text{ de acordo com } \overrightarrow{k} \\ 0 & \text{caso contrário} \end{cases}$$

Assim, o vetor $\overrightarrow{k_i}$ é uma referência ao estado de $\overrightarrow{k}$, no qual o nodo k_i está ativo na rede e todos os outros não. A motivação de construir este vetor é para que seja possível, utilizando o esquema de ponderação **TF-IDF**, somar as contribuições individuais dos termos de indexação e de $\overrightarrow{k_i}$ (BAEZA-YATES; RIBEIRO-NETO, 2013), permitindo considerar, isoladamente, a contribuição do termo k_i.

Utilizando estes conceitos, pode-se definir:

$$P(q|\overrightarrow{k}) = \begin{cases} \frac{w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,q}^2}} & \text{se } \overrightarrow{k} = \overrightarrow{k_i} \quad \wedge \quad on(i, \overrightarrow{q}) = 1 \\ 0 & \text{caso contrário} \end{cases} \tag{2.25}$$

$$P(d_j|\overrightarrow{k}) = \begin{cases} \frac{w_{i,j}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2}} & \text{se } \overrightarrow{k} = \overrightarrow{k_i} \quad \wedge \quad on(i, \overrightarrow{d_j}) = 1 \\ 0 & \text{caso contrário} \end{cases} \tag{2.26}$$

Então, o ranqueamento dos documentos recuperados, definidos por $P(d_j|q)$, é similar ao ordenamento gerado pelo VSM, uma vez que as probabilidades condicionais estão especificadas com base neste modelo clássico.

Portanto, observa-se que o modelo BNM não impõe custos significativos adicionais em sua modelagem (BAEZA-YATES; RIBEIRO-NETO, 2013), além de fornecer um formalismo claro para combinar diferentes fontes de evidência em suporte a um dado documento. Esta combinação de evidências, conforme mencionado em (TURTLE; CROFT, 1991), pode ser usada para melhorar os resultados produzidos pelo modelo.

2.1.3.6 PageRank

No passado, a RI na *Web* era fundamentalmente baseada nos modelos de RI apresentados até então. As principais diferenças relativas à RI clássica eram: (a) as coleções eram compostas por páginas *Web*; (b) as páginas precisavam ser coletadas e (c) as coleções eram significativamente maiores (BAEZA-YATES; RIBEIRO-NETO, 2013). Estas diferenças, portanto, eram suficientes para fazer com que estes modelos de RI apresentassem resultados consideravelmente inferiores em relação à coleções formadas por texto não estruturado, sem a topologia complexa e a grande

escala da *Web*. A partir deste problema, inúmeros estudos foram realizados sobre a utilização da estrutura sintática e semântica das páginas *Web* para melhoria da eficácia nos resultados de uma consulta, destacando-se a utilização de *links* presentes na estrutura das páginas no intuito modificar o ranqueamento gerado pelos modelos e melhorar a acurácia das pesquisas na *Web*. É exatamente a partir desta ideia que surge o modelo de RI PageRank (BRIN; PAGE, 1998).

A ideia básica do PageRank é introduzir uma noção de autoridade de página, que difere da ideia intuitiva de conteúdo de uma página. Tal medida de autoridade é encontrada apenas na estrutura topológica da *Web*. Em particular, a autoridade de uma página p depende do número de *links* que apontam para ela e da autoridade da página q que cita p com um link direto. Além disso, supõe-se que citações seletivas de q a p forneçam mais contribuição para a pontuação de p do que citações uniformes. Assim, o PageRank PR_p de p é calculado levando em consideração o conjunto de páginas $pag[p]$ que apontam para p (BIANCHINI; GORI; SCARSELLI, 2005). Ou seja:

$$PR_p = (1 - d) + d \sum_{q \in pag[p]} \frac{PR_q}{h_q} \tag{2.27}$$

onde d é um fator de amortecimento e h_q é o *outdegree* de q, isto é, o número de *links* saindo de q. Analisando a equação do PageRank, pode-se perceber que uma página terá uma maior relevância se ela for tomada a partir de várias referências de outras páginas. De forma análoga, caso páginas com maior PageRank possuírem *links* apontando para outras páginas, estas terão maiores chances de melhorar sua relevância no algoritmo. Isto evita o efeito colateral de aumento na relevância causado quando várias páginas de menor relevância apontam para uma outra determinada página. Portanto, para aumentar o PageRank de determinado documento, é necessário que mais *links* estejam apontando para ele; porém, estes *links* apontados devem ter boa relevância para contribuir no ranqueamento. É importante ressaltar, também, que diversas melhorias foram propostas para garantir que os *links* relevantes tenham qualidade e que sejam semanticamente relacionados ao conteúdo da página, tornando-se possível, inclusive, aplicar o PageRank fora do domínio da *Web* (GLEICH, 2015).

O modelo PageRank é independente de consulta, isto é, é um modelo de RI que recebe um grafo de páginas *Web* como entrada e atribui uma classificação a cada documento que pode especificar a autoridade relativa desse documento na *Web*. Para testar o PageRank de forma dependente de consultas, deve-se utilizá-lo em conjunto com algum outro modelo de RI. Neste trabalho, foi implementado uma versão do PageRank que leva em conta a pontuação dependente de consulta gerada pelo modelo BM25 (vide Subseção 2.1.3.3) na geração da função de ranqueamento final:

$$sim(d_j, q) = PR(d_j) * BM25(d_j, q) \tag{2.28}$$

Desta forma, o PageRank pode ser utilizado e comparado com os demais modelos de RI utilizados neste trabalho. Vale ressaltar, por fim, que o PageRank de um determinado documento pode ser computado em tempo de indexação, uma vez que não há dependência em relação à consultas, tornando o processo de busca mais eficiente.

2.1.4 Agregação de Modelos

Todos os modelos de RI discutidos nas Seções 2.1.1, 2.1.2 e 2.1.3 estão atrelados à caracterização de funções de similaridade para cada documento $dj \in D$ em relação a uma consulta q desejada, isto é, são modelos para criação de ranqueamentos π em relação a D. No entanto, existem outras tarefas complementares que podem fornecer melhorias na qualidade dos ranqueamentos gerados, como é o caso das técnicas de agregação. Na tarefa de agregação de ranqueamentos, tem-se que, para uma consulta q, são geradas k listas ranqueadas com base na coleção de documentos: $\sum = \{\pi_i | \pi \in \prod,\ i = 1, \ldots, k\}$, onde $\prod$ é o conjunto de todos os ranqueamentos em D. A agregação de ranqueamentos, portanto, recebe uma consulta q e diferentes ranqueamentos $\pi_i \in \sum$, como entrada, e produzem um novo ranqueamento π como resultado da combinação destes diferentes ranqueamentos.

Neste sentido, a agregação de ranqueamento é, na verdade, um processo de combinação dos resultados de diferentes modelos de RI em um único modelo resultante, que pode ser melhor do que qualquer um dos modelos inicialmente gerados (LI, 2014). Desta forma, pode-se considerar a agregação de ranqueamento como um conjunto de técnicas para construção de modelos *ensemble* para RI. Além disso, um modelo *ensemble*, conforme já discutido, produz uma função de ranqueamento que combina funções de similaridade de distintos modelos de RI, podendo ser úteis para melhorar a eficácia dos resultados obtidos.

O restante desta seção é organizado como se segue. A Subseção 2.1.4.1 descreve o método de agregação chamado *Borda Count* e, por fim, a Subseção 2.1.4.2 descreve o método de agregação *Markov Chain*.

2.1.4.1 *Borda Count*

O método de agregação *Borda Count* (BC), proposto por (ASLAM; MONTAGUE, 2001), combina os ranqueamentos de entrada baseando-se na posição dos documentos nos *rankings* de entrada. Especificamente, no saída do modelo, os documentos são ordenados de acordo com o número de documentos que foram ranqueados abaixo deles nos ranqueamentos de entrada (LI, 2011). Desta forma, o método prioriza aqueles modelos que mais vezes apareceram no topo dos *rankings* de entrada.

A pontuação final dos documentos após a agregação é dada por S_D e pode ser calculada como:

$$S_D = F(\sum) = \sum_{i=1}^{k} S_i \tag{2.29}$$

$$S_i \equiv \begin{bmatrix} s_{i,1} \\ \cdots \\ s_{i,j} \\ \cdots \\ s_{i,n} \end{bmatrix} \tag{2.30}$$

$$s_{i,j} = n - \pi_i(j) \tag{2.31}$$

onde $s_{i,j}$ denota o número de documentos ranqueados atrás do documento d_j no *ranking* básico π_i, $\pi_i(j)$ corresponde à posição do documento d_j no *ranking* $\pi_i(j)$ e n é o número de documentos.

O método BC, segundo (LI, 2014), pode ser visto como um método que associa um vetor de k valores (dados por $n - \pi_i(j)$) para cada documento e ordena-os por meio da norma L_1 dos vetores. Neste sentido, pode-se facilmente encontrar outras variantes do BC, de acordo com a forma de processar estes vetores. Desta forma, percebe-se que o método de agregação BC é extremamente simples, eficiente e pode ser uma boa solução para geração de *ensembles*.

2.1.4.2 *Markov Chain*

O método de agregação *Markov Chain* (MC) assume que há uma Cadeia de Markov nos documentos a serem ranqueados. Na Estatística, uma Cadeia de Markov é um caso particular de um processo estocástico com estados discretos com a propriedade de que a distribuição de probabilidade do próximo estado depende apenas do estado atual e não da sequência de eventos que o precederam. Neste sentido, os estados anteriores são irrelevantes para a predição dos estados seguintes, desde que o estado atual seja conhecido (NORRIS, 1998).

Dentro do contexto de agregação de ranqueamentos, pode-se modelar as relações de precedência dos documentos nos ranqueamentos básicos por meio de uma matriz de probabilidades de transição em uma Cadeia de Markov. Uma distribuição de probabilidade então é gerada para definir tais probabilidades de transição entre os documentos. (DUH; KIRCHHOFF, 2008) propuseram diferentes métodos para construir as probabilidades de transição na Cadeia de Markov de documentos e, desta forma, construir o ranqueamento agregado. Neste trabalho, é utilizada a técnica de agregação chamada MC4.

No método MC4, se o estado corrente é o documento d_i, então o próximo estado é decidido como se segue. Um documento d_j é selecionado uniformemente dentre o conjunto formado pela união de todos os documentos D. Se $j \succ_k i$, onde $j \succ_k i$ significa que j é melhor

ranqueado que i no *ranking* k, em uma quantidade maior de *rankings* básicos k, então vá para d_j na cadeia; caso contrário, continue em d_i. A matriz de probabilidades P pode ser definida como:

$$P \equiv (p(i,j))_{n \times n} \tag{2.32}$$

$$p(i,j) = \begin{cases} \frac{1}{n}, & q(i,j) > q(j,i) \\ \frac{n-m}{n}, & j = i \\ 0, & \textit{caso contrário} \end{cases} \tag{2.33}$$

onde n é o número de documentos, $m = ||\{j|q(i,j) > q(j,i)\}||$ e q pode ser definido como:

$$q(i,j) = \sum_k q_k(i,j) \tag{2.34}$$

$$q_k(i,j) = \begin{cases} 1, & j \succ_k i \\ 0, & \textit{caso contrário} \end{cases} \tag{2.35}$$

A partir desta distribuição, o ranqueamento agregado é construído por meio da Cadeia de Markov (DUH; KIRCHHOFF, 2008). O método MC4 descrito, tal como o BC (vide Subseção 2.1.4.1), baseia suas decisões finais de ranqueamento por meio de técnicas de votação da maioria. Na verdade, os métodos tratam todos os ranqueamentos de entrada igualmente e dão altas pontuações àqueles documentos com classificação alta na maioria das entradas. No entanto, a suposição de peso uniforme pode não ser válida em muitos casos práticos. De fato, as listas de ranqueamento geradas por diferentes modelos de RI podem ter diferentes precisões e confiabilidade.

2.1.5 Arquivo Invertido

Em geral, sistemas de RI armazenam sua coleção de documentos em memória secundária, no chamado *repositório central*. Os documentos deste repositório, por sua vez, precisam passar por um processo de indexação para que as operações de recuperação e de ranqueamento sejam efetuadas. Um índice, de acordo com (SILVA; COTA, 2004), é uma estrutura de dados antiga e muito utilizada para a rápida recuperação de informação, correspondendo a uma coleção de palavras selecionadas (ou conceitos) ligadas por meio de apontadores às informações relacionadas a ela. Uma das principais estruturas de dados responsáveis pela indexação de uma coleção, segundo (SILVA; COTA, 2004) é o Arquivo Invertido ou Índice Invertido.

Um Arquivo Invertido consiste em um mecanismo orientado a palavras para a indexação de uma coleção de texto, a fim de acelerar a tarefa da busca (JANEIRO, 2017). A estrutura do

índice é composta por dois elementos: o *vocabulário* (também chamado de dicionário) e as *ocorrências*. O vocabulário constitui o conjunto de palavras-chave da coleção. Para cada palavra-chave do vocabulário, o arquivo invertido armazena uma lista de ocorrências de documentos que contêm esta palavra. Por esta razão, ele é chamado de Arquivo Invertido, pois é possível reconstruir o texto por meio deste índice (BAEZA-YATES; RIBEIRO-NETO, 2013), desde que as posições no texto de cada uma das ocorrências tenham sido armazenadas.

Para um melhor entendimento, a Figura 2.7 ilustra um Arquivo Invertido a partir da coleção especificada na Tabela 2.5.

Tabela 2.5 – Exemplo de uma coleção textual (SILVA; COTA, 2004)

Documento	Texto
Doc 1	a casa da mãe
Doc 2	casa da mãe
Doc 3	mãe da casa
Doc 4	a da casa da mãe
Doc 5	a casa da mãe a casa mãe

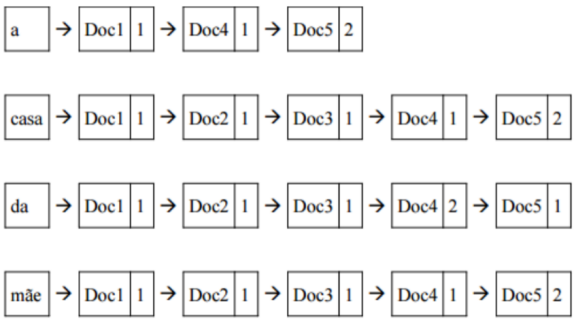

Figura 2.7 – Arquivo Invertido tradicional confeccionado para a coleção de documentos da Tabela 2.5 (SILVA; COTA, 2004)

De acordo com a Figura 2.7, para cada palavra-chave do vocabulário, há uma lista de ocorrências (documento e frequência) associadas a ele. Dessa forma, observa-se que um Arquivo Invertido facilita a pesquisa por termos dentro de uma determinada coleção de documentos e, também, possibilita a obtenção eficaz de diferentes estatísticas, como as medidas TF e IDF. No entanto, esta construção abordada na Figura 2.7 é extremamente básica e simplificada, e não representa um caso real de utilização de Arquivos Invertidos. Normalmente, em sistemas de RI, antes da construção do Arquivo Invertido, é feita uma etapa de análise e processamento do texto, onde é aplicada a *tokenização* (dividir o texto em palavras, chamadas de *tokens*), são removidas as *stopwords*, é aplicado *stemming* e até mesmo operações de filtragem (remoção de palavras específicas do domínio da aplicação). Desta forma, o Arquivo Invertido não pode ser mais utilizado para reconstituir o texto e não é mais recomendado para localizar proximidade entre termos, e sim para realizar consultas por meio de palavras-chave.

Ademais, um sistema de RI deve se preocupar com a *eficiência* de seu índice. Embora a eficiência possa parecer uma questão secundária em relação à eficácia, ela raramente pode ser negligenciada no projeto de um sistema de RI. O processo de construção e manutenção de Arquivos Invertidos é um procedimento relativamente simples quando é possível armazenar o texto e o índice em memória principal. No entanto, intuitivamente, estes modelos só são eficazes enquanto a estrutura indexada cabe na memória o que, na prática, é um caso raro (BAEZA-YATES; RIBEIRO-NETO, 2013). Uma das formas de se resolver este problema é criar índices parciais I_i, ou segmentos, obtidos após a indexação de um subconjunto dos documentos da coleção.

Depois de toda coleção ser indexada, vários índices parciais I_i existem em disco. Estes índices podem ser combinados de maneira hierárquica, como mostra a Figura 2.8 (em tal figura, os retângulos representam os índices parciais e os círculos representam operações de união). Por exemplo, os índices I_1 e I_2 são combinados pela operação de união 1 para obter o índice $I_{1..2}$. Esse processo de combinação pode continuar até que, possivelmente, haja apenas um índice que compreende toda a coleção.

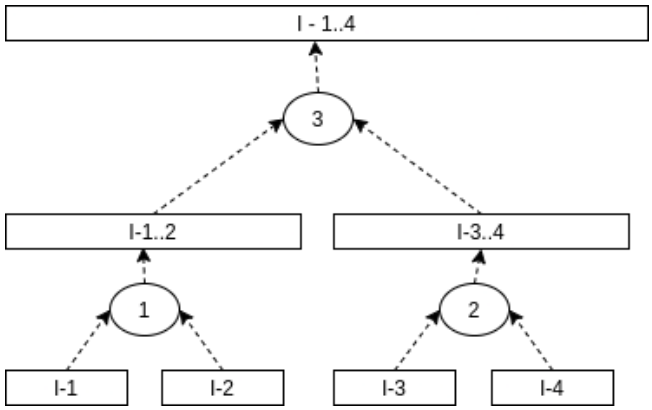

Figura 2.8 – Combinação dos índices parciais de forma binária. (BAEZA-YATES; RIBEIRO-NETO, 2013)

A fusão de dois índices consiste em combinar os vocabulários, unindo a lista de ocorrências de cada palavra (BAEZA-YATES; RIBEIRO-NETO, 2013). Considerando uma memória principal de tamanho M disponível e uma coleção de n documentos, uma possível estratégia é gerar $\frac{n}{M}$ índices parciais e combiná-los a um custo de entrada e saída de $O(nlog(\frac{n}{M}))$, pois há $log_2(\frac{n}{M})$ níveis hierárquicos de índices parciais. Mais de dois índices podem ser combinados ao mesmo tempo, melhorando ainda mais a eficiência desta estratégia. Por fim, há de se ressaltar que, em muitos casos, não é viável manter um único segmento de índice para determinada coleção, isto é, pode ser mais interessante manter índices parciais de tamanho reduzido do que um único índice completo que, em alguns casos, é muito grande e de difícil manipulação.

Esta abordagem para criação de índices facilita, também, o processo de atualização desta estrutura de dados. Para adicionar novos documentos, basta criar um índice parcial que compreende os documentos adicionados e, em seguida, combiná-lo com um índice parcial(total)

da coleção. A exclusão de um documento, por sua vez, deve ser feita por meio de uma passada sobre o Arquivo Invertido para remover ocorrências dos termos do documento.

O Arquivo Invertido foi utilizado no ATRI no intuito de prover uma boa eficiência mediante às consultas realizadas. Quando um usuário adiciona documentos em uma coleção, internamente, é gerado um Arquivo Invertido parcial contendo todas as ocorrências dos termos em cada documento adicionado. Desta forma, o índice do ambiente é escalável e, apesar da adição de uma coleção de documentos demorar algum tempo em casos mais robustos, o usuário ganha um tempo considerável na realização de consultas.

2.2 Trabalhos Relacionados

O trabalho proposto pode ser visto sob diferentes perspectivas de pesquisa. Desta forma, são apresentados trabalhos relacionados sob duas diferentes direções que, de certa forma, estão relacionadas com os objetivos deste trabalho: construção de ferramentas amigáveis para cálculo de similaridade entre documentos, e construção de ambientes para pesquisa e comparação empírica de modelos do estado da arte para RI.

Especificamente quanto à construção de ferramentas amigáveis para cálculo de similaridade entre documentos, (JANEIRO, 2017) propôs uma ferramenta, denominada MatchUp, contemplando os modelos clássicos (vide Subseção 2.1.2) e o EBM (vide Subseção 2.1.3.1) de RI, que, por sua vez, estão presentes também neste trabalho. Destaca-se, também, a interface amigável proposta pela ferramenta (vide Figura 2.9) para que usuários não especializados possam usá-la em situações em que se deseja saber a proximidade de um determinado documento ou um conjunto de termos de interesse em relação a um conjunto de documentos pertencentes a uma coleção. (JANEIRO, 2017) conduziu um estudo comparativo que sugeriu que o modelo VSM, utilizando o esquema de ponderação **TF-IDF** Normalização Logarítimica (TF) e Frequência Inversa (IDF), apresentou os melhores resultados de uma forma geral, sendo incluído como o modelo *default* de execução da ferramenta. Para a experimentação prática proposta, os experimentos foram realizados considerando uma pequena coleção de teste relativa a restaurantes locais, atingindo ótimos valores da métrica **F1**, sendo 75% para os 3 primeiros documentos retornados, e $83,3\%$, para os 7 primeiros documentos retornados. Como pontos negativos da ferramenta, nota-se que o esquema de indexação proposto foi construído inteiramente em memória principal e, por isso, ele suporta apenas pequenas coleções; ademais, não há nenhuma possibilidade de customização das funções de ranqueamento além da parametrização dos modelos de RI implementados.

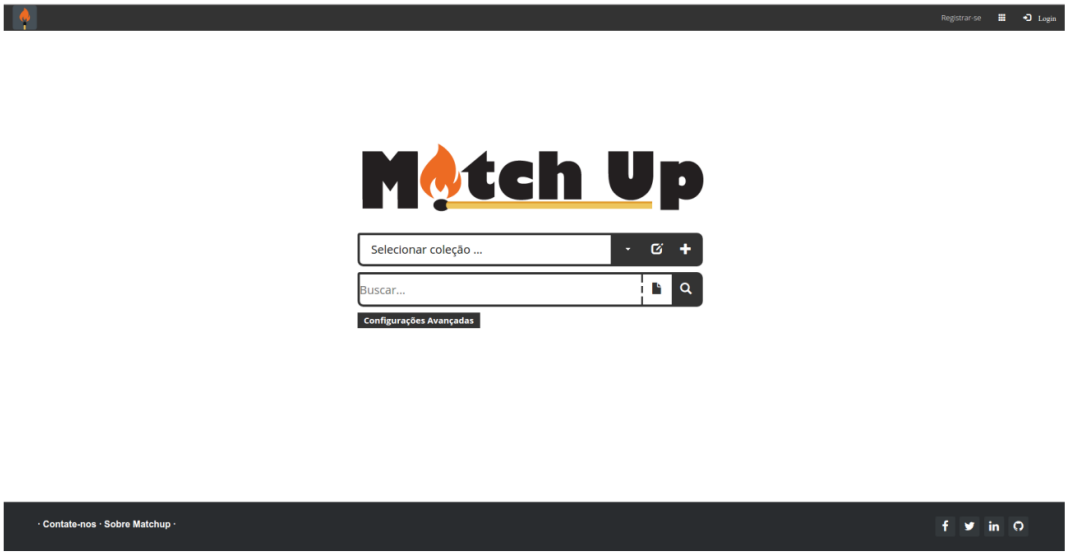

Figura 2.9 – Interface da tela principal do MatchUp (JANEIRO, 2017)

Outra ferramenta para cálculo de similaridade entre documentos é o Quepid (QUEPID, 2021), que tem como objetivo ser uma ferramenta para teste e melhoria da relevância em pesquisas dos mecanismos de busca Solr[7] e ElasticSearch[8]. A ferramenta visa, por meio de uma interface amigável (vide Figura 2.10), ajudar o usuário a configurar e ajustar os parâmetros e a função de similaridade do seu mecanismo de busca e, por meio da coleta de métricas de eficácia mediante a consultas, permite analisar e comparar a eficácia dos resultados obtidos. Diferentemente do MatchUp (JANEIRO, 2017) e deste trabalho, a ferramenta tem o viés puramente comercial e é limitada a coleções de documentos preestabelecidas da ferramenta ou importadas pelos mecanismos de busca suportados. Além disso, a ferramenta possui limitações quanto aos modelos de RI disponíveis para customização, estando totalmente limitada às opções oferecidas pelo mecanismo de busca associado às coleções preestabelecidas de documentos.

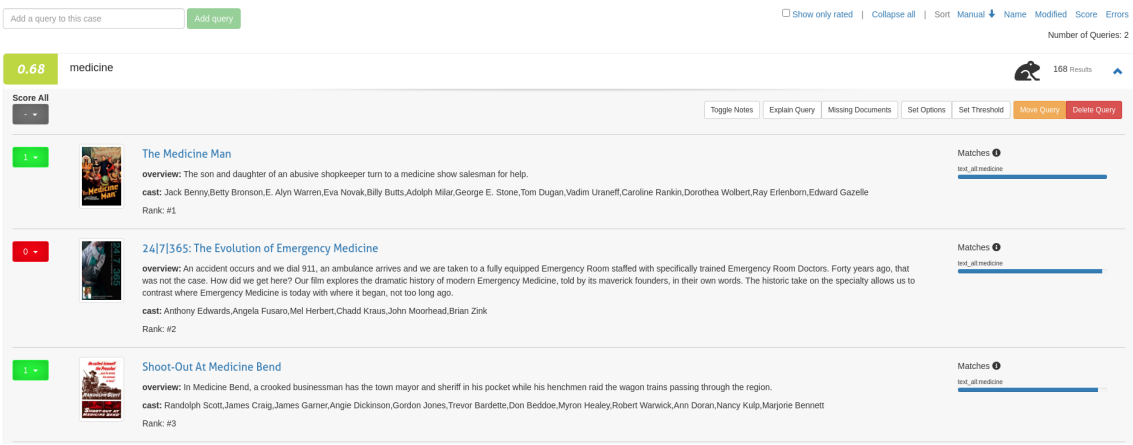

Figura 2.10 – Interface da tela principal de consulta do Quepid.

[7] Disponível em <https://solr.apache.org/>
[8] Disponível em <https://www.elastic.co/>

Por fim, quanto aos trabalhos relacionados à construção de ambientes para pesquisa e comparação empírica de modelos no estado da arte para RI, destaca-se o ambiente denominado Anserini[9] (YANG; FANG; LIN, 2017). Em linhas gerais, Anserini é um ambiente para pesquisa reprodutível em RI, que visa preencher a lacuna entre a pesquisa acadêmica de RI e a prática de construção de sistemas de busca do mundo real. Proposto a partir de um estudo de reprodutibilidade sobre diferentes sistemas de RI (LIN et al., 2016), a ferramenta atualmente provê *baselines* para comparação de eficácia de diferentes funções de similaridade considerando diversas coleções de documentos e utilizando como base diferentes modelos de RI, o que, desta forma, tem grande visibilidade acadêmica. No entanto, a ferramenta não apresenta, claramente, formas de customização e criação de funções de similaridade, nem mesmo uma interface amigável para realização de consultas e coleta de métricas, tornando seu uso extremamente restrito a usuários avançados e pesquisadores científicos.

Este trabalho, em diversos fatores, difere-se dos demais já mencionados. Tal como (JANEIRO, 2017), este trabalho visa construir um ambiente de RI, denominado ATRI, que permita o cálculo de similaridade entre documentos e consultas por meio de uma interface amigável. No entanto, diferentemente de (JANEIRO, 2017), este trabalho visa apresentar uma alta variedade de modelos de RI em um ambiente que suporte grandes coleções, e tal como (QUEPID, 2021), coletar métricas de eficácia mediante a consultas por meio do próprio ambiente, visando facilitar a tarefa de teste das funções de similaridade suportadas pelo ATRI. Ademais, este trabalho visa, tal como em (YANG; FANG; LIN, 2017), criar *baselines* considerando diferentes coleções padronizadas de acordo com os modelos de RI disponíveis na ferramenta, tornando-o mais atrativo para a comunidade científica de RI.

[9] Disponível em <https://github.com/castorini/Anserini>

3 Ambiente Experimental Proposto

Como já mencionado, este trabalho possui, como objetivo geral, a proposta e o desenvolvimento do ATRI[1], um ambiente de RI para cálculo de similaridade entre consultas e coleções de documentos, possibilitando a geração de ranqueamentos de relevância para cada consulta desejada, podendo ser um determinado documento ou um conjunto de termos de interesse do usuário. Para tanto, o ATRI permite a utilização dos seguintes modelos de RI como base para o cálculo de similaridade: Booleano, Vetorial, Probabilístico, BM25, Redes de Crença, Booleano Estendido, Vetorial Generalizado, DFRee, PL2 e PageRank.

Este capítulo apresenta o próprio ATRI, estando delineado da seguinte forma. A Seção 3.1 descreve a arquitetura de funcionamento do ATRI. As Seções 3.2 e 3.3 descrevem a parametrização necessária para a efetivação, respectivamente, das consultas e das funções de similaridade utilizadas pelos modelos de RI do ATRI. Por fim, a Seção 3.4 apresenta a interface de utilização do ambiente.

3.1 Arquitetura de Funcionamento do ATRI

O ATRI possibilita ao usuário a realização de consultas, podendo ser termos ou um determinado documento de seu interesse, em uma coleção de documentos, por meio de uma interface amigável que facilita a entrada dos dados necessários e que permite a intervenção do usuário, se for o caso, no estabelecimento de características de funcionalidades quanto aos modelos de RI a serem usados. A Figura 3.1 ilustra a arquitetura de funcionamento do ambiente em alto nível.

A partir da Figura 3.1, pode-se entender um fluxo comum de utilização do ambiente:

- Passo 01: uma coleção de documentos é especificada pelo usuário via interface. Para cada coleção especificada, documentos podem ser encaminhados pelo usuário e são enviados para o componente de "Formatação de Dados".
- Passo 02: a partir dos documentos originais obtidos no Passo 01, um processo de formatação nos dados é feito no intuito de transformar documentos de qualquer extensão suportada pelo ambiente em documentos indexáveis pelo mesmo. O ATRI torna possível, também, a adição de transformadores próprios criados pelos usuários para tornar documentos de extensões desconhecidas indexáveis.
- Passo 03: a partir dos documentos originais formatados (Passo 02), é ativado o módulo “Motor de busca”, composto por diferentes componentes fundamentais para o funciona-

[1] O ATRI encontra-se publicamente disponível no repositório <https://github.com/atri-search/atri>

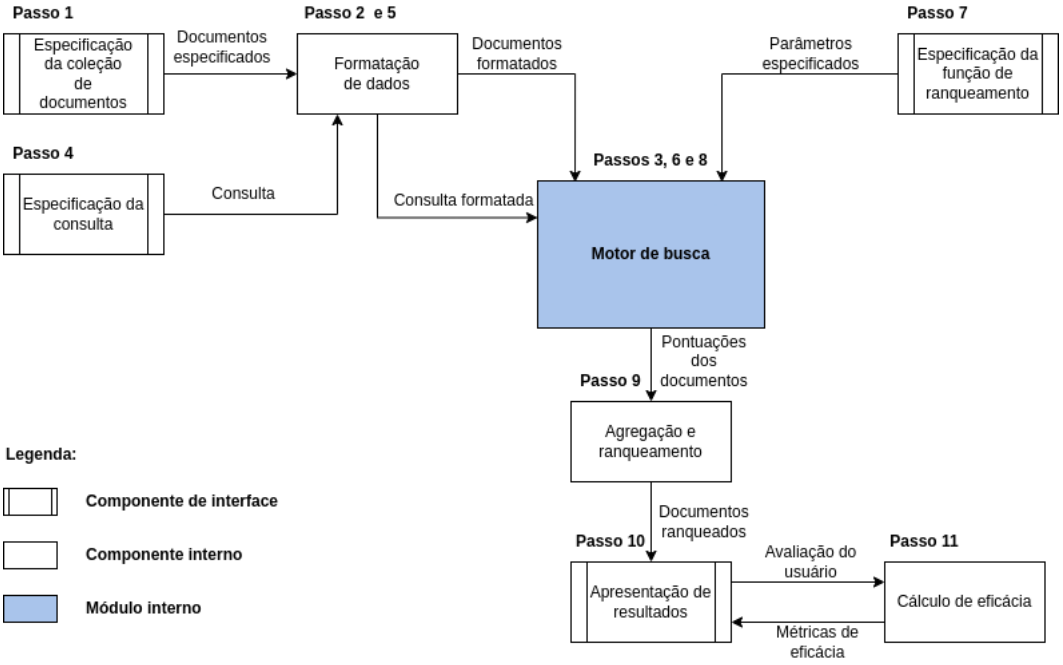

Figura 3.1 – Arquitetura de Funcionamento do ATRI

mento do ambiente. Neste caso, o motor de busca realizará um processo de indexação dos documentos formatados, criando um segmento de índice físico.

- Passo 04: uma consulta é especificada pelo usuário via interface, podendo ser um conjunto de termos ou um determinado documento. Esta consulta é encaminhada ao componente “Formatação de Dados".

- Passo 05: de forma análoga ao Passo 02, a consulta passa por uma formatação de dados para adequação no ambiente. Neste caso, também, é possível adicionar transformadores próprios criados pelos usuários para adequar diferentes tipos de consulta.

- Passo 06: a consulta formatada (Passo 05) é inserida no "Motor de busca", que a executará no Passo 08 de acordo com o índice previamente construído.

- Passo 07: a função de ranqueamento é especificada pelo usuário via interface, considerando os diferentes modelos de RI (vide Subseções 2.1.2 e 2.1.3) e as técnicas de agregação de ranqueamentos implementadas neste trabalho (vide Subseção 2.1.4). A especificação de características particulares, como esquemas de ponderação, parâmetros numéricos dos modelos de RI e esquemas de agregação de ranqueamentos, é feita por meio de uma opção de consulta avançada do ambiente. Vale ressaltar que a função de ranqueamento com os melhores resultados, de acordo com os experimentos realizados neste trabalho, foi definida como *default* do ambiente. A função de ranqueamento especificada é enviada para o "Motor de busca".

- Passo 08: considerando o Índice Invertido gerado no Passo 03, a consulta tratada no Passo 06 e a função de ranqueamento e suas características especificadas no Passo 07, é realizado

um processo de cálculo de similaridade entre a consulta tratada e os documentos tratados da coleção representados pelo Índice Invertido. Os documentos recuperados, via cálculo de similaridade, são associados a pontuações de relevância e encaminhados ao componente "Agregação e Ranqueamento" (Passo 09).

- Passo 09: os documentos recuperados, obtidos no Passo 08, são ordenados do mais similar para o menos similar em relação à consulta especificada pelo usuário. Caso algum modelo de agregação tenha sido definido no Passo 07, a agregação de ranqueamentos também é executada, gerando um ranqueamento final. Os documentos similares ordenados, produzidos nesse passo, são passados para o componente de interface "Apresentação de resultados" (Passo 10).

- Passo 10: os documentos similares ordenados, obtidos no Passo 09, são apresentados ao usuário, como resultados da consulta especificada, por meio da interface do ATRI. Nesta etapa, é permitido ao usuário definir os documentos ranqueados como "Relevante"ou "Não Relevante", no intuito de produzir (Passo 11) uma pontuação de eficácia por meio de métricas padronizadas.

- Passo 11: o *feedback* recebido no Passo 10 é utilizado para computação da eficácia do ranqueamento produzido. Esta pontuação é gerada por meio de métricas padronizadas e o resultado é enviado para a interface de "Apresentação de Resultados"(Passo 10).

Como pode ser visto, um módulo de destaque apresentado na Figura 3.1 é o "Motor de busca"(Passos 03, 06 e 08), responsável por indexar, remover e processar documentos, além de permitir a realização de consultas no ambiente. Este módulo pode ser expandido e apresentado em detalhes, como mostra a Figura 3.2, que exibe os submódulos e componentes fundamentais deste módulo.

A partir da Figura 3.2, destaca-se três fluxos principais de utilização do módulo "Motor de Busca": inserção de documentos (sufixo **a** nos passos), processamento de consultas (sufixo **b** nos passos) e remoção de documentos (sufixo **c** nos passos). O fluxo de inserção de documentos pode ser entendido por meio dos passos:

- Passo 1.a: os documentos inseridos no "Motor de busca"(Passo 03 da arquitetura geral exibida na Figura 3.1) são compostos por campos[2] que, por sua vez, podem ser de dois tipos: armazenados ou indexados. Os campos armazenados são armazenados como informação descritiva do documento e não são utilizados no processo de recuperação. Desta forma, campos armazenados são passados diretamente para o Módulo de Escrita (Passo 3.a). Por outro lado, os campos indexados são utilizados no processo de recuperação e, por isso, são passados para o "Módulo de Análise e Processamento de Texto"(Passo 2.a).

[2] Um campo, para o ATRI, é formado por uma sequência de *bytes* que, por sua vez, representam o conteúdo textual daquele campo.

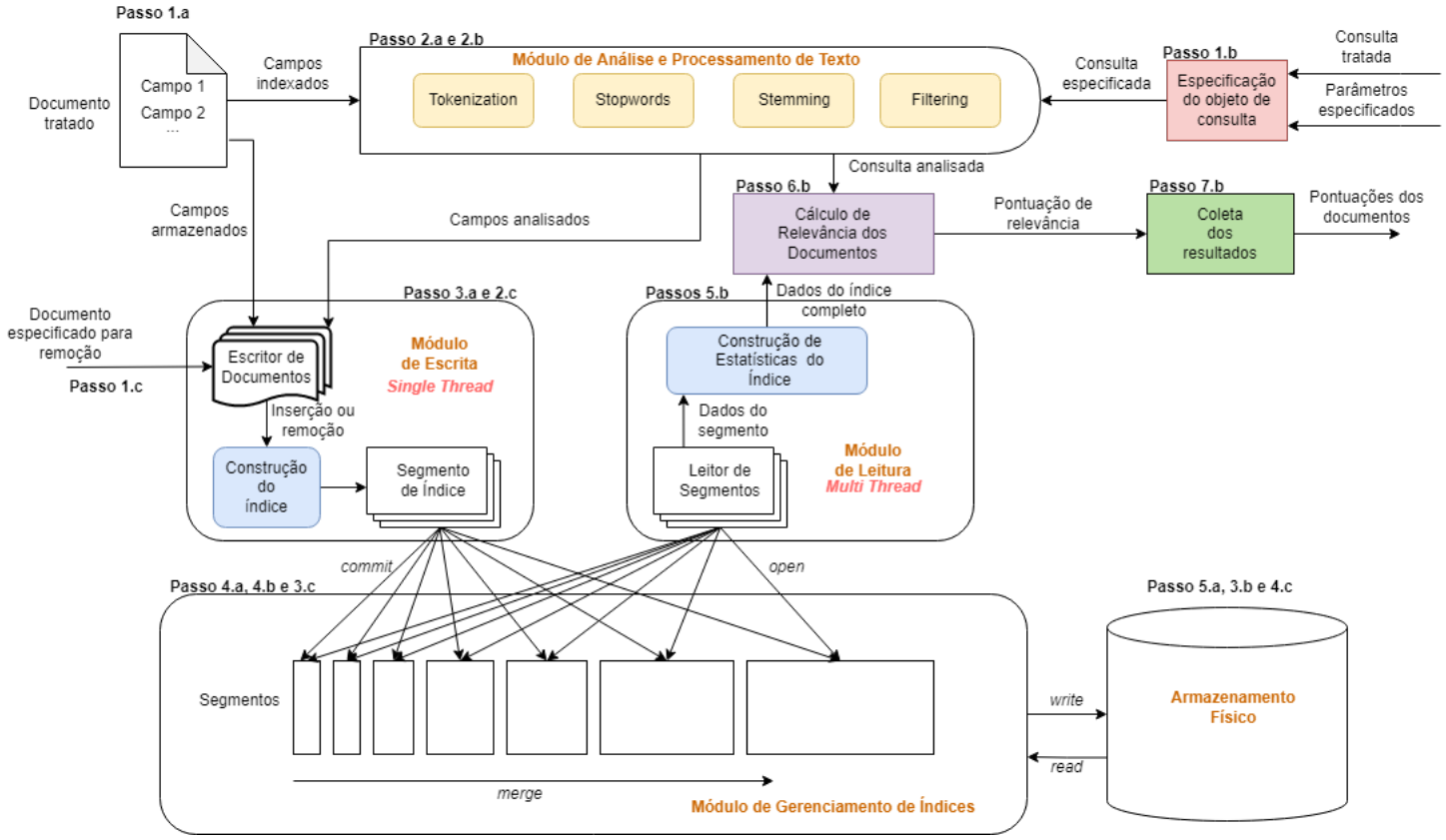

Figura 3.2 – Arquitetura de Funcionamento do Módulo "Motor de busca"(vide Figura 3.1)

- Passo 2.a: os campos indexados (Passo 1.a) são processados com operações de tokenização, *stemming* e remoção de *stopwords*. A aplicação de *filtros* é opcional e deve ser configurada no ambiente. Os campos processados são enviados para o "Módulo de Escrita"(Passo 3.a).

- Passo 3.a: os campos armazenados e indexados são utilizados para criação do índice invertido parcial, ou segmento de índice, dos documentos que estão sendo inseridos. Para manter a consistência e isolamento do índice, o módulo deve ser *single thread*, isto é, apenas um documento atualiza o índice invertido em um dado instante de tempo. O segmento de índice gerado é confirmado (*commit*) e enviado para o "Módulo de Gerenciamento de Índices"(Passo 4.a).

- Passo 4.a: o segmento de índice gerado (Passo 3.a) é combinado com outros segmentos existentes, caso seja necessário. O índice atualizado é enviado para escrita (*write*) em disco.

- Passo 5.a: o índice parcial gerado no passo anterior é armazenado em um repositório central físico, garantindo a persistência dos dados indexados.

De acordo com o Passo 4.a, os segmentos de índice podem ser combinados. Esta combinação (*merge*) dos segmentos de índice é feita de tal forma a evitar que muitos segmentos pequenos sejam formados, o que é indesejável considerando que há um *overhead* de performance para manter cada segmento de índice e, portanto, muitos segmentos pequenos trazem lentidão à

aplicação. Desta forma, a combinação de segmentos é feita com índices considerados pequenos pelo ambiente. Segmentos de índice grandes normalmente não são combinados, pois a operação pode ser demorada e sem vantagens claras de utilização, uma vez que o segmento gerado pode não caber em memória principal.

Para fazer consultas em documentos de uma coleção, o seguinte fluxo é formado no "Motor de busca":

- Passo 1.b: a consulta tratada e os parâmetros da consulta especificados (Passos 05 e 07, respectivamente, da Figura 3.1) são enviados ao componente de Especificação do Objeto de Consulta que, por sua vez, cria a representação da consulta que será utilizada dentro do "Motor de busca". O objeto de consulta especificado é passado ao "Módulo de Análise e Processamento de Texto"(Passo 2.b).

- Passo 2.b: similar ao Passo 2.a, os campos textuais da consulta são tratados com tokenização, *stemming* e remoção de *stopwords*. A consulta analisada é enviada para o componente de "Cálculo de Relevância dos Documentos"(Passo 6.b).

- Passo 3.b: O índice físico armazenado é aberto para leitura para ser utilizado posteriormente no processo de consulta pelo "Módulo de Gerenciamento de Índices"(Passo 4.b).

- Passo 4.b: o "Módulo de Gerenciamento de Índices"realiza a leitura do índice do repositório central (Passo 3.b) para que, posteriormente, execute a combinação dos diferentes segmentos de índice de acordo com o objeto de consulta que será especificado pelo "Módulo de Leitura"(Passo 5.b).

- Passo 5.b: o leitor de segmentos do "Módulo de Leitura"é ativado, realizando a leitura dos segmentos de índice com base no objeto de consulta especificado. O segmentos são então abertos (*open*) no "Módulo de Gerenciamento de Índices"e os índices são combinados (Passo 4.b). A partir da representação em memória obtida, diferentes estatísticas são coletadas referentes aos documentos da coleção candidatos para a consulta. Note que esta etapa é *multi-thread*, isto é, o índice pode ser lido por diferentes processos no mesmo instante de tempo. Por fim, as estatísticas são enviadas ao componente de "Cálculo de Relevância dos Documentos"(Passo 6.b).

- Passo 6.b: a consulta analisada (Passo 2.b) e os dados referentes aos documentos (Passo 5.b) são utilizados para realizar o cálculo de similaridade entre documentos. Nesta etapa, a função de ranqueamento previamente especificada no ambiente é utilizada para computar pontuações para cada documento. A pontuação de relevância gerada para os documentos é enviada ao componente de "Coleta de Resultados"(Passo 7.b).

- Passo 7.b: as pontuações de relevância obtidas do passo anterior são recebidas e as computações finais para a obtenção da pontuação final dos documentos são executadas por este

componente. A operação de normalização pelo tamanho dos documentos, por exemplo, pode ser realizada nesta etapa. Por fim, as pontuações dos documentos são enviadas para o ambiente.

A partir do fluxo de consulta do "Motor de busca", os documentos são devidamente ordenados e, caso seja especificado, uma operação de agregação pode acontecer considerando distintos ranqueamentos gerados, como mostra o Passo 09 da arquitetura do ambiente exibida na Figura 3.1.

Por fim, a última operação suportada pelo "Motor de busca" é a remoção de documentos do índice (vide Figura 3.2) . Esta operação é extremamente importante para manutenção de índices previamente gerados. Seu fluxo é extremamente simples:

- Passo 1.c : os documentos especificados para remoção são passados para o "Módulo de Escrita do ambiente" (Passo 2.c).
- Passo 2.c : os documentos especificados são inseridos no "Módulo de Escrita" e uma operação de escrita no índice é iniciada. Por questões de eficiência, “excluir” um documento simplesmente significa adicioná-lo a uma lista de documentos excluídos que será armazenada com o índice. Quando o índice for lido, o ambiente sabe que não deve retornar os documentos excluídos nos resultados. Esta lista é enviada ao "Módulo de Gerenciamento de Índices"(Passo 3.c).
- Passo 3.c : a lista de arquivos removidos é atualizada e enviada ao "Armazenamento Físico"(Passo 4.c).
- Passo 4.c : a lista de arquivos atualizada no Passo 3.c anterior é escrita (*write*) no disco físico, garantindo a exclusão bem sucedida dos documentos.

Note que, no fluxo especificado acima, a operação de exclusão é puramente virtual, isto é, não há espaço em disco liberado; o documento desejado é acrescido a uma lista de documentos excluídos. No entanto, caso uma operação de mesclagem seja executada (*merge*), o índice será refeito e as remoções serão fisicamente excluídas, podendo ser retiradas da lista de documentos excluídos e mantendo o "Motor de busca" com boa eficiência.

3.2 Configuração da Consulta no ATRI

Conforme apresentado na Figura 3.2, o Passo 1.b da arquitetura de funcionamento do "Motor de busca" para execução de consultas corresponde à especificação do objeto de consulta, dado a consulta do usuário tratada e alguns parâmetros adicionais que podem ser especificados pelo usuário via interface. Para a efetivação da consulta no ATRI, portanto, duas pré-condições fundamentais precisam ser satisfeitas no ambiente:

- Índice Invertido: corresponde à coleção de documentos previamente indexada para busca (vide Subseção 2.1.5);
- Consulta: corresponde à especificação dos termos relativos à consulta, especificados pelo usuário via interface no Passo 04 da Figura 3.1; os termos podem ser especificados livremente pelo usuário ou pertencer a um determinado documento.

A partir das pré-condições fundamentais apresentadas, o usuário pode configurar manualmente, por meio da interface, parâmetros de configuração para a execução de cada consulta, conforme mostra a Tabela 3.1.

Tabela 3.1 – Parâmetros de consulta no ATRI

Parâmetro	Valores	Descrição
fieldname	$f \in F_j$	Nome do campo de busca
query	$\{"or", "and"\}$	Modo de operação da consulta

A especificação do objeto de consulta no ambiente deve ser feita considerando um campo indexado pelos documentos (vide Subseção 3.1), podendo ser especificado pelo usuário por meio do parâmetro *fieldname* (vide Tabela 3.1). Os valores aceitos pelo parâmetro *fieldname* são campos $f \in F_j$, onde F_j corresponde ao conjunto de campos indexados por um dado documento d_j. Atualmente, no ATRI, a busca só pode ser realizada tomando em conta um único campo indexado. Abaixo, há um exemplo prático da especificação do campo de consulta no ambiente:

```
{
  "fieldname": "body", (1)
}
```

Conforme a especificação (1), o campo configurado para a realização das consultas é o corpo (do inglês, *body*) dos documentos. No ATRI, a escolha do campo *body* é *default* para a realização de consultas.

Uma consulta em um sistema de RI deve realizar o cálculo de similaridade para cada documento individualmente e, posteriormente, realizar a ordenação por relevância dos mesmos. Este processo, contudo, é intuitivamente ineficiente quando o processo de consulta envolve grandes coleções de documentos. Uma forma de reduzir o custo computacional das consultas, portanto, é filtrar os documentos por meio dos termos de interesse especificados na consulta do usuário. Essa filtragem, no ATRI, é feita utilizando um modo de operação que pode ser especificado pelo usuário por meio do parâmetro *query* (vide Tabela 3.1), correspondendo a um conectivo booleano (AND ou OR) que conecta os termos de interesse da consulta. Por meio do modo de operação, apenas os documentos que satisfazem a expressão lógica do objeto de consulta formado serão considerados candidatos no cálculo de similaridade, reduzindo os custos computacionais da consulta. Note, portanto, que a especificação de conectivos lógicos entre os termos da consulta não é restrita a modelos booleanos, como BM e EBM, no ATRI. Abaixo, há

a continuação do exemplo anterior com a especificação do modo de operação da consulta no ambiente:

```
{
  "fieldname": "body",
  "query": "or" (2)
}
```

Conforme a especificação (2), o conectivo lógico configurado para o modo de operação é o OR. No ATRI, a escolha deste conectivo é *default* para a execução de consultas.

3.3 Configuração da Função de Ranqueamento no ATRI

Conforme apresentado na Figura 3.1, o Passo 07 da arquitetura de funcionamento do ATRI corresponde à especificação de modelos de RI e suas características, no intuito de construir uma função de ranqueamento personalizada pelo usuário. Os modelos de RI, conforme já apresentado, definem estratégias distintas para a busca de documentos considerados relevantes, em uma coleção, dada uma consulta (vide Subseções 2.1.2 e 2.1.3). Dessa forma, o ATRI provê um ambiente que permite a configuração da função de ranqueamento via interface.

De uma forma geral, a função de ranqueamento pode ser definida utilizando um ou mais modelos de RI presentes no ambiente. Cada modelo deve ser especificado juntamente com seus parâmetros formais obrigatórios (vide Subseções 2.1.2 e 2.1.3). Caso apenas um modelo seja especificado, a consulta será executada diretamente. No entanto, caso mais de um modelo for especificado, a consulta será executada em ambos e, posteriormente, será executada uma operação de agregação dos resultados obtidos (vide Subseção 2.1.4). A Tabela 3.2 exibe os parâmetros de configuração para a execução de cada modelo de RI no ambiente.

Tabela 3.2 – Parâmetros de Função de Ranqueamento no ATRI

Modelo de RI	Parâmetros obrigatórios
boolean	
vector_space	*tf, idf*
probabilistic	
bm25	*k1, b*
pl2	*c*
dfree	
belief_network	*tf, idf*
extended_boolean	*tf, idf*
generalized_vector_space	*tf, idf*

A parametrização da função de ranqueamento no ATRI parte de um parâmetro obrigatório chamado *similarity*, que corresponde a um ou mais modelos de RI (conforme apresentado na Tabela 3.2), seguido dos parâmetros formais obrigatórios de cada modelo escolhido. Um exemplo

de configuração da função de ranqueamento utilizando apenas um modelo de RI pode ser ilustrada na configuração abaixo:

```
{
  "fieldname": "body",
  "query": "or",
  "similarity": "vector_space", (3)
  "tf": "frequency", (4)
  "idf": "inverse_frequency" (5)
}
```

A especificação acima exibe um exemplo de configuração válida para a função de ranqueamento no ambiente. Em tal exemplo, as configurações de campo de consulta e modo de operação seguem a especificação mostrada na Subseção 3.2; ademais, a função de ranqueamento (3) é definida utilizando apenas o modelo VSM, juntamente com seus parâmetros formais TF (4) e IDF (5), que são definidos como, respectivamente, Frequência e Frequência Inversa (vide Subseção 2.1.2.2).

A especificação da função de ranqueamento considerando um único modelo de RI é análoga para os outros modelos definidos no ATRI. Neste caso, há apenas variações na definição dos parâmetros formais de cada um deles.

Para a criação da função de ranqueamento considerando mais de um modelo de RI, cada modelo deve ser especificado juntamente com seus parâmetros formais obrigatórios (vide Subseções 2.1.2 e 2.1.3) e, adicionalmente, um método de agregação deve ser utilizado. Um exemplo de configuração da função de ranqueamento utilizando três modelos de RI pode ser ilustrada na configuração abaixo:

```
{
  "aggregation": "borda_count", (6)
  "fieldname": "body",
  "query": "or",
  "similarity": ["vector_space", "bm25", "extended_boolean"], (7)
  "idf": "inverse_frequency",
  "tf": "frequency",
  "b": 0.75, (8)
  "k1": 1.2, (9)
  "p": 3.0 (10)
}
```

Na configuração acima, três modelos de RI são passados por meio de uma lista na configuração de similaridade (7) e, para cada modelo, seus parâmetros formais são especificados. O modelo VSM é configurado conforme o exemplo anterior. O modelo BM25, por sua vez, é configurado com parâmetros $b = 0.75$ (8) e $k1 = 1.2$ (9). Já o modelo EBM é configurado com $p = 3$ (10). Por fim, o método de agregação (6) utilizado é o BordaCount (vide Subseção 2.1.4).

A configuração é feita de forma análoga para qualquer combinação de modelos de RI. Note que, atualmente, esta abordagem possui duas grandes limitações:

- não há a possibilidade de combinar dois modelos iguais, mas com diferentes parâmetros;
- não há a possibilidade de definir diferentes esquemas de ponderação TF-IDF para diferentes modelos.

Embora haja limitações, esta forma de configuração da função de ranqueamento é bastante expressiva e provê ao usuário muita flexibilidade para configuração de funções personalizadas, além de permitir a criação de *plugins*, isto é, extensões que permitem a intervenção do próprio usuário no processo de busca.

3.4 Interface de Funcionamento do ATRI

Conforme mencionado na Seção 3.1, existem diferentes especificações que devem ser realizadas pelo usuário via interface do ambiente. Nesta seção, é abordada a interface gráfica do ATRI que possibilita ao usuário a execução dos diferentes passos no fluxo de utilização do ambiente.

Em relação aos Passos 04 e 07, conforme mostra a Figura 3.1, as especificações referentes à consulta e à função de ranqueamento devem ser realizadas pelo usuário via interface. Para tanto, foi desenvolvida a tela principal do ambiente, ilustrada na Figura 3.3.

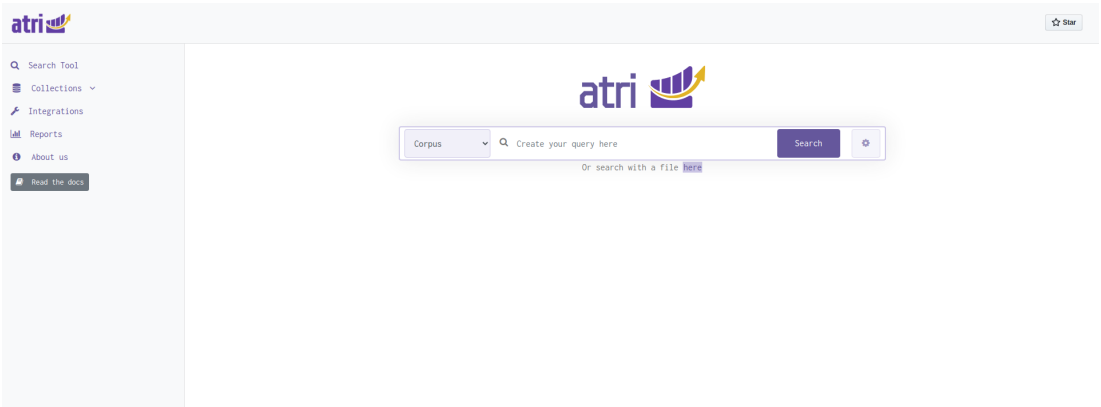

Figura 3.3 – Interface da Tela Principal do ATRI

De acordo com a Figura 3.3, pode-ser observar que a tela principal do ATRI dispõe de três campos principais para realização de uma consulta. São eles:

- *Corpus*: por meio deste campo, o usuário seleciona uma coleção previamente criada e indexada;

- *Search*: por meio deste campo, o usuário especifica uma consulta, em linguagem natural ou por meio de um documento, a ser feita sobre a coleção selecionada; esta especificação corresponde ao Passo 04 da arquitetura de funcionamento do ambiente (vide Figura 3.1);
- Configurações: por meio deste campo (representado com o botão de engrenagem), é possível estabelecer as especificações da função de ranqueamento mencionadas, no Passo 07 da arquitetura de funcionamento.

Além dos campos apresentados, a tela da Figura 3.3 apresenta um menu lateral de navegação para outras páginas relevantes do ambiente. Dentre as páginas presentes, há uma sessão para criação e manutenção de coleções (representado por "*Collections*"). Ao ser criada, uma coleção pode ser configurada de diversas formas, como mostra a Figura 3.4.

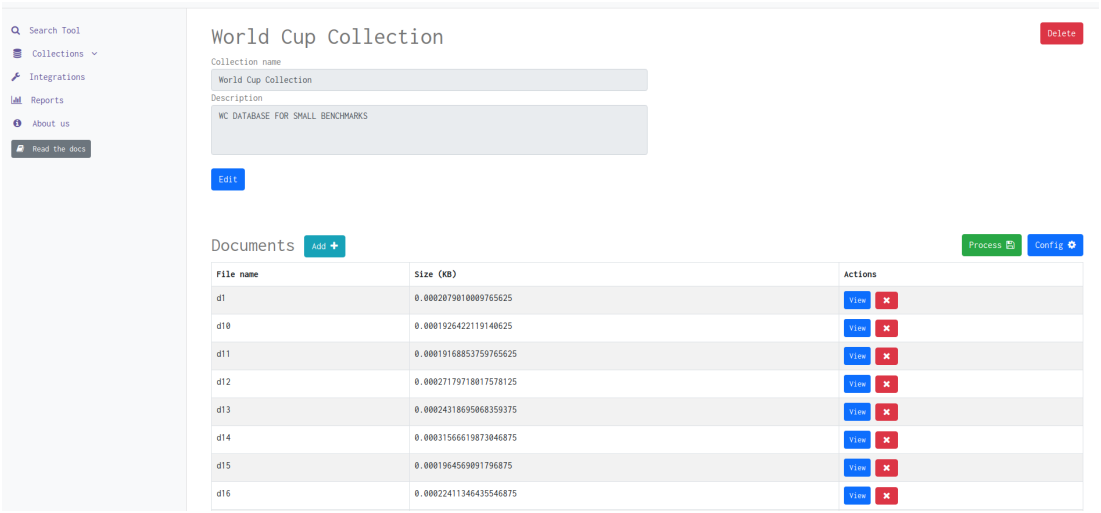

Figura 3.4 – Interface da Gerência de uma coleção no ATRI

A Figura 3.4 exibe uma lista de todos os seus documentos de uma coleção e apresenta todas as operações possíveis de serem realizadas nas coleções do ambiente, por meio dos botões:

- *Edit*: edita os metadados da coleção;
- *Delete*: deleta completamente a coleção e seus documentos;
- *Add*: adiciona documentos à coleção;
- *Process*: computa o índice invertido da coleção;
- *Config*: permite adição de configurações referentes ao processo de indexação da coleção.

Em relação à etapa de configuração da coleção, o processo de geração do Índice Invertido no ambiente pode ser definido como incremental ou total, isto é, o usuário pode definir se irá indexar todos os documentos da coleção (reconstrução do índice) ou se apenas os novos documentos inseridos serão indexados. A indexação incremental provê ao ambiente escalabilidade

para indexar grandes coleções de documentos, enquanto a indexação total provê um mecanismo para melhorar a eficiência dos processos de recuperação para coleções menores ou de reconstrução do índice em caso da remoção de muitos documentos.

Por fim, os Passos 10 e 11 da arquitetura de funcionamento (vide Figura 3.1) são responsáveis por exibir ao usuário final os resultados de uma consulta e, também, permitir a obtenção de métricas de eficácia a partir do *feedback* emitido pelo usuário mediante ao resultado da consulta realizada. Para tanto, foi desenvolvida a tela de resultados do ambiente, ilustrada na Figura 3.5.

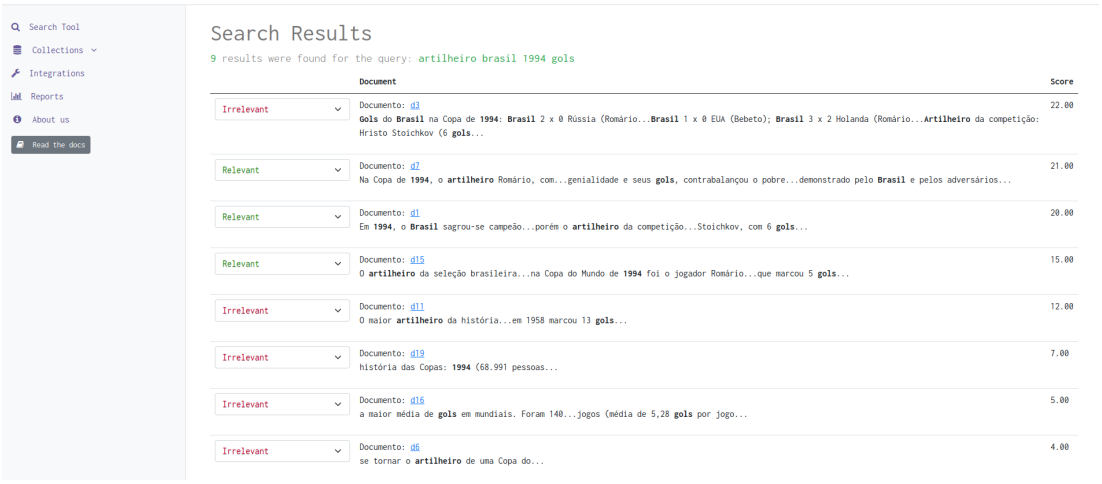

Figura 3.5 – Interface dos resultados de uma consulta no ATRI

De acordo com a Figura 3.5, pode-ser observar que a tela de resultados do ATRI exibe ao usuário: a lista de documentos ordenados por relevância, a pontuação de cada documento, uma sumarização do texto de cada documento, e um campo opcional para o usuário emitir seu *feedback* de cada resultado gerado. Em relação à geração de métricas de eficácia, o usuário deve especificar o gabarito da busca rotulando os documentos como *Relevant* ou *Irrelevant* e, a partir do gabarito especificado, as métricas de eficácia são obtidas e exibidas na interface.

4 Experimentação Prática

Neste capítulo, são apresentados e analisados os experimentos realizados no ambiente ATRI, seguindo a arquitetura proposta na Seção 3.1. A Subseção 4.1 descreve as métricas de avaliação de eficácia utilizadas. A Subseção 4.2 descreve as coleções de teste utilizadas e descreve os experimentos realizados. Por fim, a Subseção 4.3 apresenta e avalia os resultados obtidos por meio dos experimentos realizados.

4.1 Métricas de Avaliação

Para avaliação da eficácia de funções de ranqueamento em um sistema de RI, diferentes métricas podem ser utilizadas, tais como a precisão, a revocação e o F1. Seja R o conjunto de documentos relevantes à consulta e seja A o conjunto dos documentos retornados por uma função de ranqueamento. Neste caso, de acordo com (BAEZA-YATES; RIBEIRO-NETO, 2013) e considerando o contexto deste trabalho, a precisão (*precision*) pode ser definida como a fração dos documentos recuperados que é relevante; ou seja:

$$\textit{precision} = \frac{|R \cap A|}{|A|} \tag{4.1}$$

Também conforme definido por (BAEZA-YATES; RIBEIRO-NETO, 2013) e aplicado ao contexto deste trabalho, revocação (*recall*) é definida como a fração dos documentos relevantes que foi recuperada; ou seja:

$$\textit{recall} = \frac{|R \cap A|}{|R|} \tag{4.2}$$

Por fim, segundo (MANNING; RAGHAVAN; SCHüTZE, 2008), F1 corresponde à média harmônica entre as métricas de precisão e de revocação; ou seja:

$$F1 = 2 \times \frac{\textit{precision} \times \textit{recall}}{\textit{precision} + \textit{recall}} \tag{4.3}$$

Uma adaptação conveniente para a obtenção das métricas apresentadas é considerar um limiar k de documentos recuperados. A precisão, por exemplo, pode ser representada pela métrica **P@k**, que traduz a precisão encontrada considerando os k primeiros documentos recuperados, utilizando uma dada função de ranqueamento. O raciocínio é análogo para revocação (**R@k**) e F1 (**F1@k**). No caso das máquinas de busca, segundo (BAEZA-YATES; RIBEIRO-NETO, 2013), é comum medir tais métricas para os primeiros 5 ou 10 documentos retornados; eles

fornecem uma avaliação da impressão do usuário sobre os resultados e baseiam-se no fato de que as pessoas raramente acessam além de uma possível segunda página de resultados.

Por fim, outra métrica de eficácia muito importante para avaliação de um sistema de RI é o *Normalized Discounted Cumulative Gain* (NDCG). Diferentemente das métricas já mencionadas, a métrica NDCG possui a vantagem de conseguir lidar com diferentes níveis de relevância, enquanto as demais são projetadas para níveis de relevância binário (relevante ou não relevante). Além disso, o NDCG já inclui em sua medida o parâmetro de limiar k que, conforme já mencionado, é particularmente apropriado para este trabalho. A métrica NDCG é baseada em outra chamada *Discounted Cumulative Gain* (DCG), cujos documentos são classificados em escalas de relevância (0 a 5, por exemplo) e o ganho do ranqueamento gerado é medido com base na ordem dos documentos. Este ganho é acumulado do topo do ranqueamento para o final, dando mais importância para documentos altamente relevantes e que aparecem no topo. Desta forma, uma possível definição matemática para a métrica DCG é:

$$DCG@k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{log_2(i+1)} \tag{4.4}$$

onde k é o corte avaliado do *ranking* e rel_i é o grau de relevância do documento que se encontra na posição i do *ranking*. O NDCG, portanto, é uma normalização da métrica DCG considerando o valor ideal do DCG, isto é, o melhor valor possível de ser alcançado no *ranking* avaliado:

$$NDCG@k = \frac{DCG@k}{IDCG@k} \tag{4.5}$$

onde $IDCG@k$ é o DCG ideal do *ranking*.

Por fim, nota-se que o NDCG@k juntamente com a métrica P@k são mais importantes no contexto da experimentação prática deste trabalho, uma vez que o objetivo é identificar funções de ranqueamento que apresentam, no topo do *ranking*, o maior número de documentos relevantes considerando as coleções avaliadas. Neste sentido, a experimentação realizada considerou a média dos valores obtidos, para cada consulta de teste e para cada coleção, das métricas P@k e NDCG@, considerando os valores $k \in \{1, 3, 5, 10, 15\}$.

4.2 Descrição dos Experimentos

Para realizar os experimentos do ATRI, considerando os modelos de RI implementados (vide Seção 2.1), foram utilizadas consultas utilizando diferentes coleções de teste padronizadas. Os modelos de RI envolvidos em tais experimentos juntamente com seus parâmetros formais são:

- BM utilizando o conectivo lógico OR;

- VSM utilizando Normalização Dupla (TF) e Frequência Inversa Suave (IDF);
- PM;
- EBM utilizando $p = 3$;
- BM25 utilizando $k_1 = 1.2$ e $b = 0.75$;
- BNM utilizando Normalização Dupla (TF) e Frequência Inversa Suave (IDF);
- GVSM utilizando Normalização Dupla (TF) e Frequência Inversa Suave (IDF);
- PL2 utilizando $c = 1$;
- DFree;
- PageRank;
- Agregação dos modelos VSM e BM25 utilizando os mesmos parâmetros individuais descritos acima e o esquema de agregação BC;
- Agregação dos modelos VSM e BM25 utilizando os mesmos parâmetros individuais descritos acima e o esquema de agregação MC.

Para a realização da análise comparativa dos resultados relativos aos processos de consulta, envolvendo as configurações mencionadas dos modelos de RI, foram utilizadas as métricas descritas na Seção 4.1. Os valores de tais métricas foram obtidos por meio da execução de consultas no próprio ambiente, considerando os documentos retornados pelo Passo 10 da arquitetura de funcionamento do ATRI (vide Figura 3.1). O conjunto de respostas ideal, isto é, o gabarito, é fornecido pelas próprias coleções de teste por meio de análises feitas por especialistas de tais coleções ou coletado manualmente para determinadas consultas.

As três primeiras coleções utilizadas para realização dos experimentos do ATRI são altamente reconhecidas pela comunidade científica. São elas: *Cystic Fibrosis* (**CF**), *Library Information Science Collection* (**LISA**) e **NPL**. A Tabela 4.1 mostra, de uma forma geral, as configurações das coleções de teste: número de documentos e de consultas de cada coleção. Apesar de relativamente pequenas, estas coleções possuem inúmeros documentos e consultas que constituem diferentes cenários extremamente úteis para avaliar a eficácia de um dado modelo de RI. Ademais, o conteúdo das coleções é bastante variado, o que torna a avaliação dos modelos bem interessante. Os conteúdos das coleções podem ser descritos como:

1. A coleção **CF** consiste em 1239 documentos publicados entre 1974 e 1979, que discutem aspectos da doença Fibrose cística, e apresenta 100 consultas com seus respectivos gabaritos;

Tabela 4.1 – Coleções de teste utilizadas para experimentação prática do ATRI

	Documentos	Consultas
CF	1239	100
LISA	5872	35
NPL	11429	93

2. A coleção **LISA** é uma coleção que contém 5872 documentos, relativos a resumos de periódicos de mais de 45 países em mais de 20 linguagens diferentes, e apresenta um conjunto com 35 consultas e seus respectivos gabaritos;

3. A coleção **NPL** constitui uma coleção com 11429 documentos, referentes a títulos e resumos de documentos e artigos, e apresenta 93 consultas com seus respectivos gabaritos.

Nota-se que, em termos de conteúdo, a coleção **CF** apresenta um tema altamente específico, o que pode causar efeitos nos resultados obtidos. As demais coleções, por sua vez, apresentam temas que são um pouco mais genéricos. Inicialmente, as consultas padronizadas pelas coleções foram consideradas como o texto integral de sua descrição, sem nenhum tipo de expansão ou pré-processamento, o que não é o ideal para obtenção dos melhores resultados.

Por fim, no intuito de avaliar um cenário de uso real do ambiente, uma nova coleção de teste composta por páginas *Web* relativas a artigos sobre o tema COVID-19 foi proposta. Tal coleção foi coletada por meio do coletor temático Yucca (JÚNIOR; REZENDE; ASSIS,), e as consultas juntamente com seus gabaritos foram definidos manualmente para este trabalho (vide Tabela 4.2).

Tabela 4.2 – Coleção de teste utilizada COVID-19 para validação de um caso de uso real do ATRI

	Documentos	Consultas
COVID-19	1275	5

Nota-se, também, que a coleção **COVID-19** viabiliza a avaliação do modelo PageRank, que não pode ser testado nas coleções da Tabela 4.1, já que elas são formadas essencialmente por documentos não estruturados (vide Subseção 2.1.3.6).

4.3 Análise dos Resultados

Nesta seção, são apresentados e analisados os resultados obtidos por meio da experimentação prática inicial realizada no ambiente ATRI, envolvendo as configurações definidas para as funções de ranqueamento. A Figura 4.1 exibe os resultados de P@k para a coleção **CF**.

Observando os resultados da Figura 4.1 referente à precisão das consultas realizadas na coleção **CF**, nota-se que:

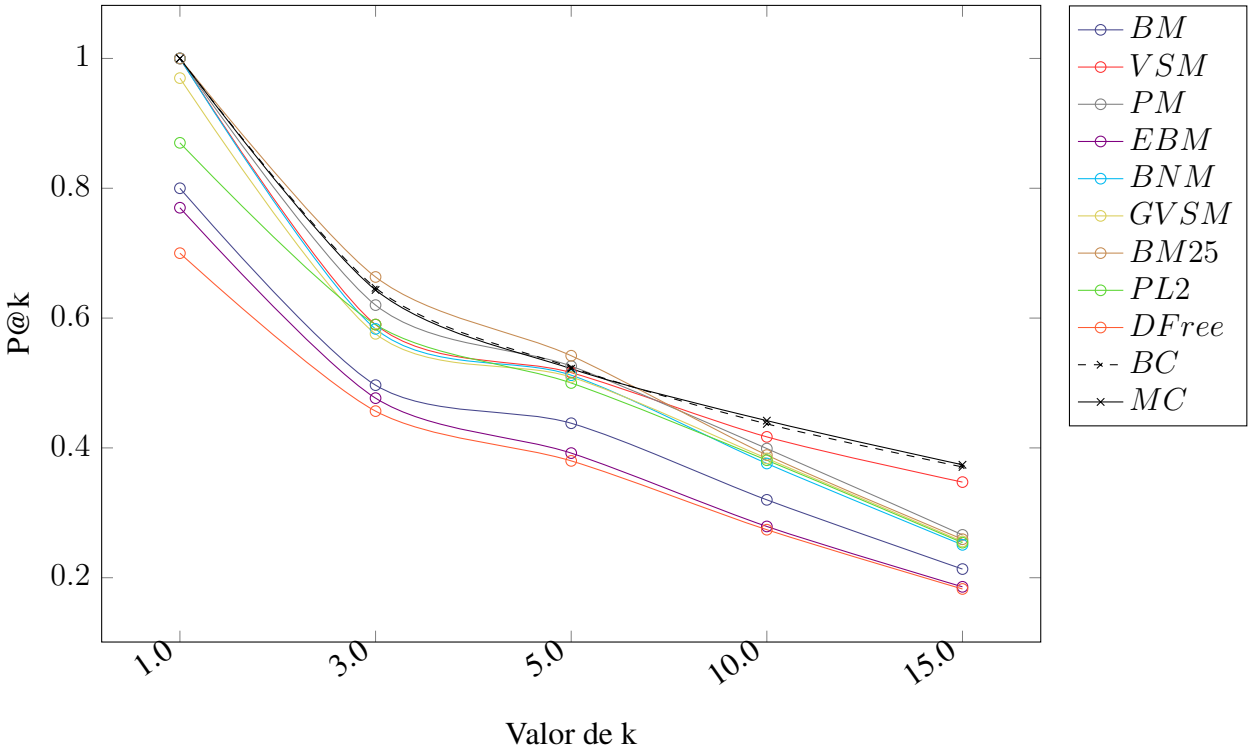

Figura 4.1 – Avaliação de P@k para a coleção CF

- de uma forma geral, os modelos comportaram-se bem considerando poucos documentos retornados, já que os valores de precisão apresentados ficaram entre 70% e 100% para $k = 1$ e entre 40% e 70% para $k = 3$;
- os modelos VSM, BM25, PM, GVSM e BNM apresentaram os melhores resultados de precisão por número de documentos retornados na coleção **CF**, destacando-se os modelos BM25 para valores pequenos de k e o VSM para valores maiores de k;
- os modelos booleanos (BM e EBM) juntamente com os baseados em DFR (PL2 e DFRee) não apresentaram bons resultados em relação aos demais já mencionados para esta coleção;
- os métodos de agregação (BC e MC) obtiveram resultados equivalentes, melhorando a precisão do VSM para valores pequenos de k e a precisão do BM25 para níveis maiores de k e, assim, tornando modelos mais equilibrados e interessantes que os individuais BM25 e VSM; destaca-se, sobretudo, o valor P@15 de cerca de 40% para estes modelos.

Da mesma forma, pode-se observar o gráfico do NDCG@k para a coleção **CF** (vide Figura 4.2). Por meio da Figura 4.2, pode-se notar que:

- diferentemente da métrica P@k, o NDCG@k distingue melhor a eficácia encontrada pelos modelos e, conforme discutido na Seção 4.1, valoriza mais os resultados relevantes recuperados no topo do ranqueamento;

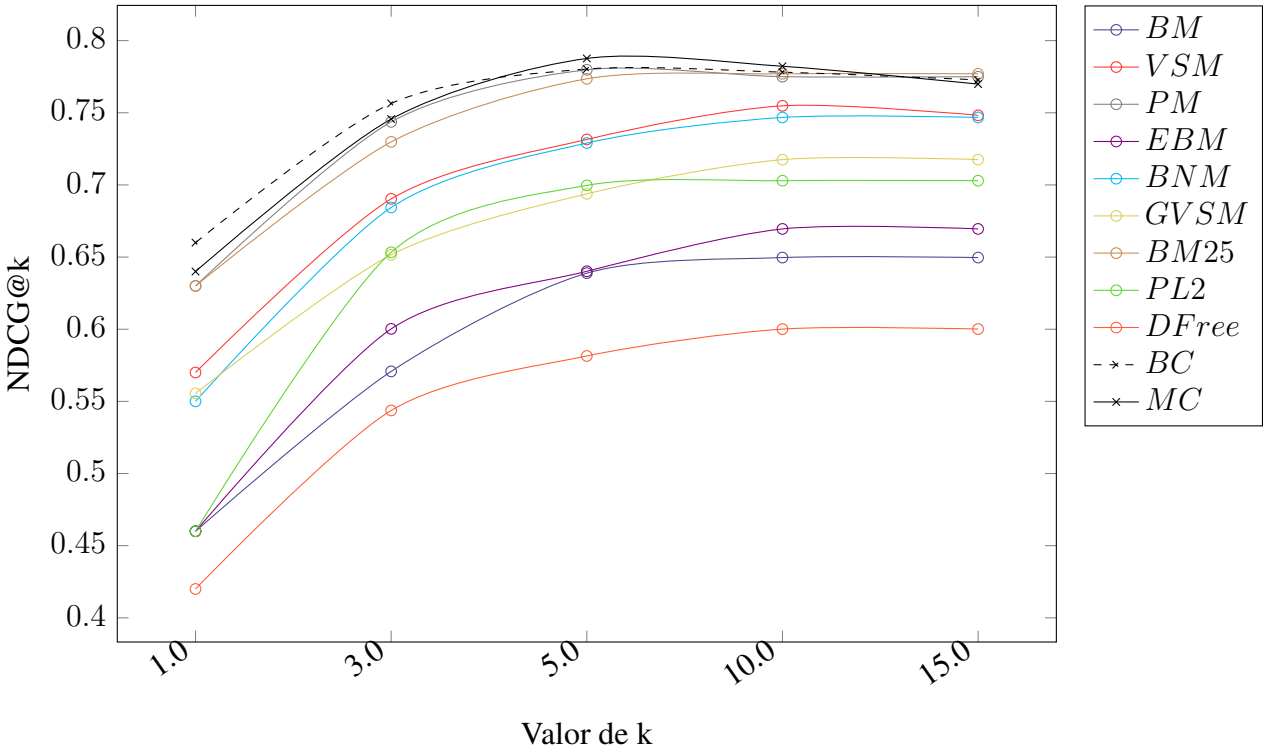

Figura 4.2 – Avaliação de NDCG@k para a coleção CF

- os modelos de RI que mais se destacaram foram os probabilísticos PM e BM25, chegando a 75% de NDCG@5;
- os modelos VSM, GVSM e BNM apresentaram boa eficácia, mas boa parte dos resultados relevantes retornados pelas consultas não estavam no topo do ranqueamento;
- os métodos de agregação (BC e MC) obtiveram os melhores resultados no geral, destacando-se a função de agregação BC com cerca de 80% de NDCG@5; além disso, por meio da análise de NDCG@k, fica perceptível que estes modelos apresentaram mais resultados relevantes no topo que os demais.

A Figura 4.3 exibe os resultados de P@k encontrados para as diferentes funções de ranqueamento avaliadas, considerando a coleção **LISA**. A partir da Figura 4.3, observa-se que:

- de uma forma geral, a precisão dos modelos foi inferior em relação à coleção **CF** (vide Figura 4.1) para os primeiros valores de k, chegando a no máximo 75% de precisão considerando P@1;
- os modelos BM25 e PL2 sobressaíram-se em relação aos demais, apresentando resultados superiores para todos os valores de k apresentados;
- os modelos booleanos EBM e BM apresentaram os piores valores de precisão para todos os valores de k;

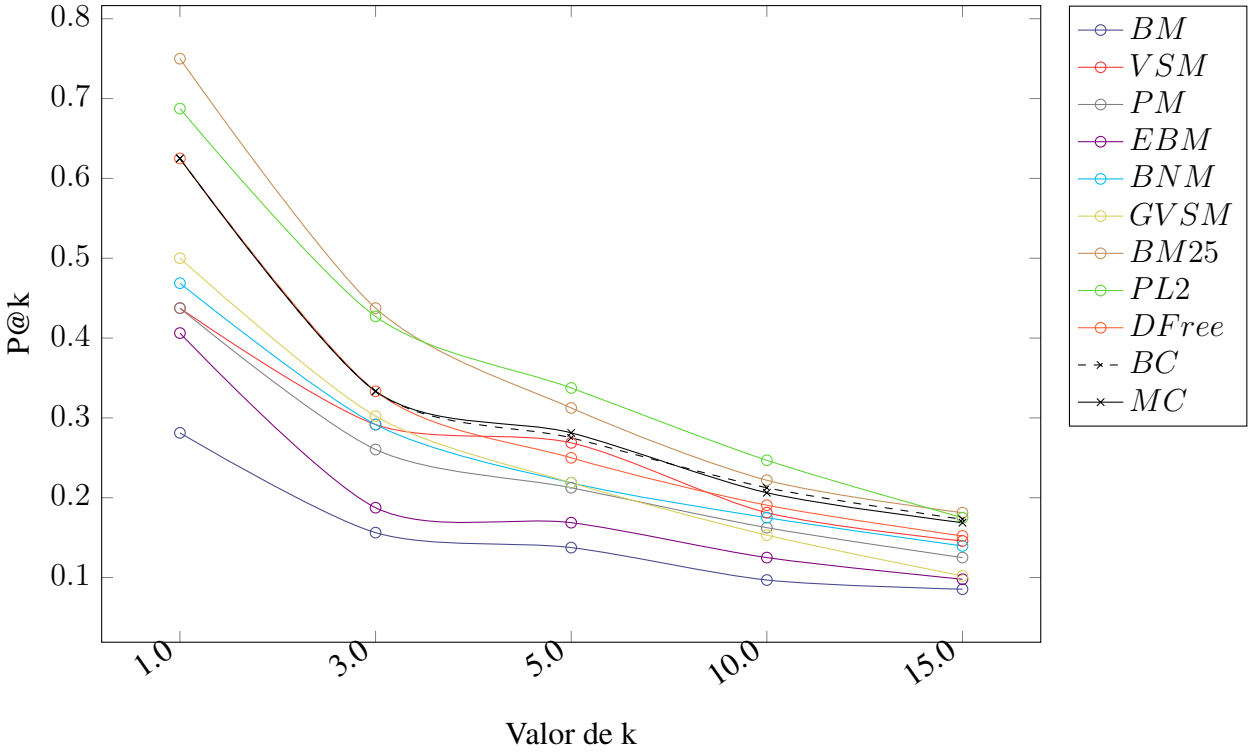

Figura 4.3 – Avaliação de P@k para a coleção LISA

- os métodos de agregação (BC e MC) obtiveram valores de precisão inferiores ao BM25, já que o modelo VSM apresentou resultados consideravelmente inferiores ao BM25 para os primeiros valores de k, tornando o modelo agregado superior ao VSM mas inferior ao BM25 para esta coleção.

Quanto à metrica NDCG@k para a coleção **LISA**, exibida na Figura 4.4, pode-se observar que:

- comprovando os resultados de P@k obtidos para a coleção **LISA**, os modelos de RI que mais se destacaram foram os modelos PL2 e BM25, chegando a valores de NDCG@5 próximos de 55%;
- o modelo VSM apresentou uma eficácia consideravelmente inferior ao BM25, impactando negativamente os *ensembles* BC e MC, formados pela agregação com o BM25;
- os modelos baseados em DFR apresentaram ótimos resultados para a coleção **LISA**, destacando-se ao modelo DFRee a obtenção de resultados menos eficazes que o PL2 mas sem a necessidade de configurar parâmetros de entrada.

Por fim, os resultados apresentados na Figura 4.5 exibem a métrica P@k referente às consultas realizadas na coleção **NPL**. A partir dos resultados obtidos, nota-se que:

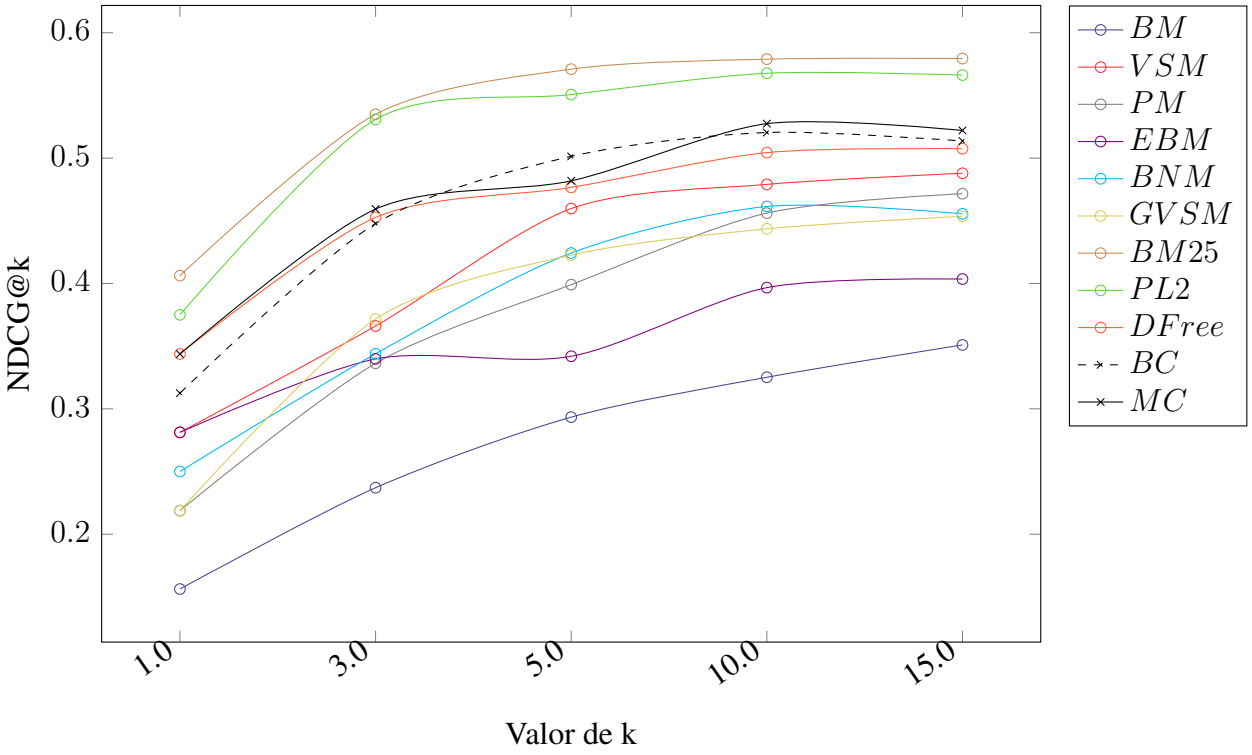

Figura 4.4 – Avaliação de NDCG@k para a coleção LISA

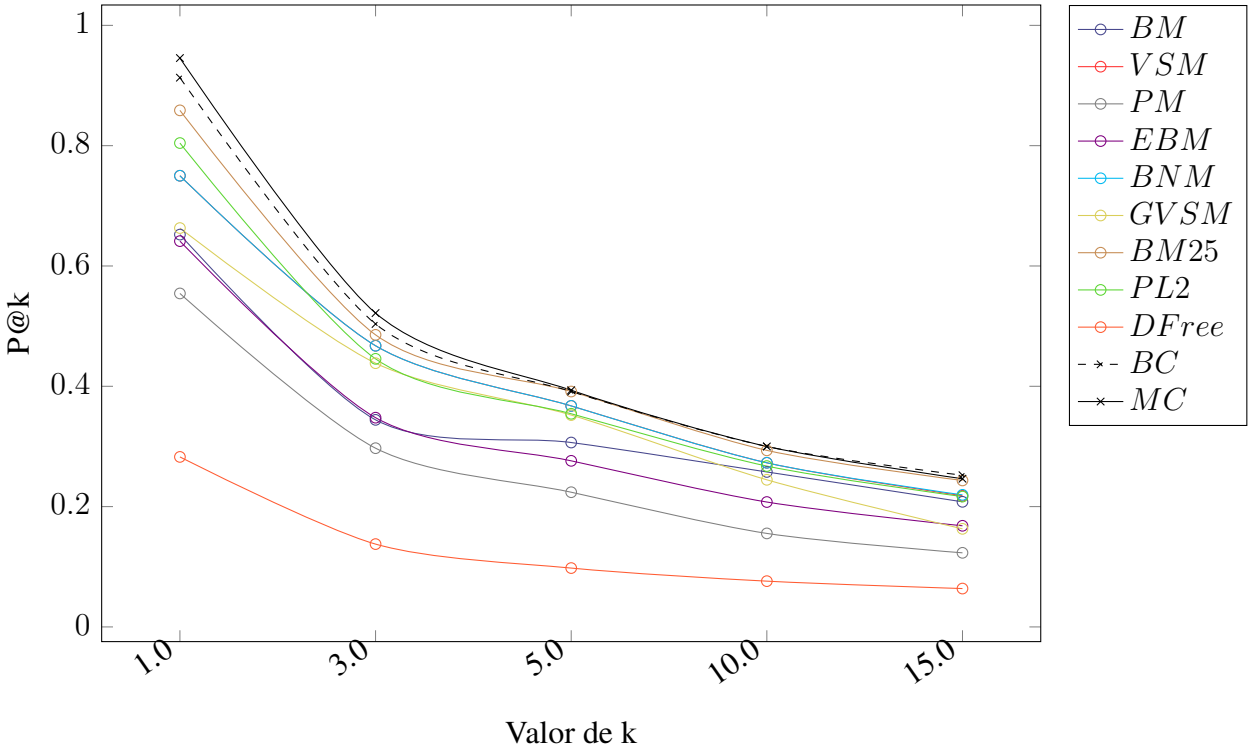

Figura 4.5 – Avaliação de P@k para a coleção NPL

- de uma forma geral, os modelos comportaram-se bem considerando poucos documentos retornados, com exceção do modelo DFree, já que os valores de precisão apresentados ficaram entre 55% e 90% para $k = 1$ e entre 35% e 50% para $k = 3$;

- os modelos BM25, PL2 e EBM apresentaram os melhores resultados de precisão por número de documentos retornados na coleção **NPL**;
- os modelos algébricos VSM, GVSM e BNM apresentaram bons resultados, mas inferiores aos modelos probabilísticos, já que o fato da coleção possuir 11429 documentos que tratam de temas ligeiramente similares pode ter gerado uma perda de especificidade nos termos da coleção, isto é, os termos passam a não descrever tão bem os tópicos de um documento específico, o que sugere descartar o peso **IDF** e, portanto, os modelos baseados em **TF-IDF** podem ter desempenho similar e não tão interessante; uma situação similar a essa ocorreu na coleção **LISA**, conforme visto na Figura 4.3;
- ambos métodos de agregação (BC e MC) obtiveram ótimos resultados, com destaque para o método MC; em linhas gerais, os modelos agregados melhoraram a precisão do BM25 e do VSM, chegando a 95% de P@1 e tornando-se novamente uma escolha adequada.

Quanto à metrica NDCG@k para a coleção **NPL**, exibida na Figura 4.6, pode-se observar que:

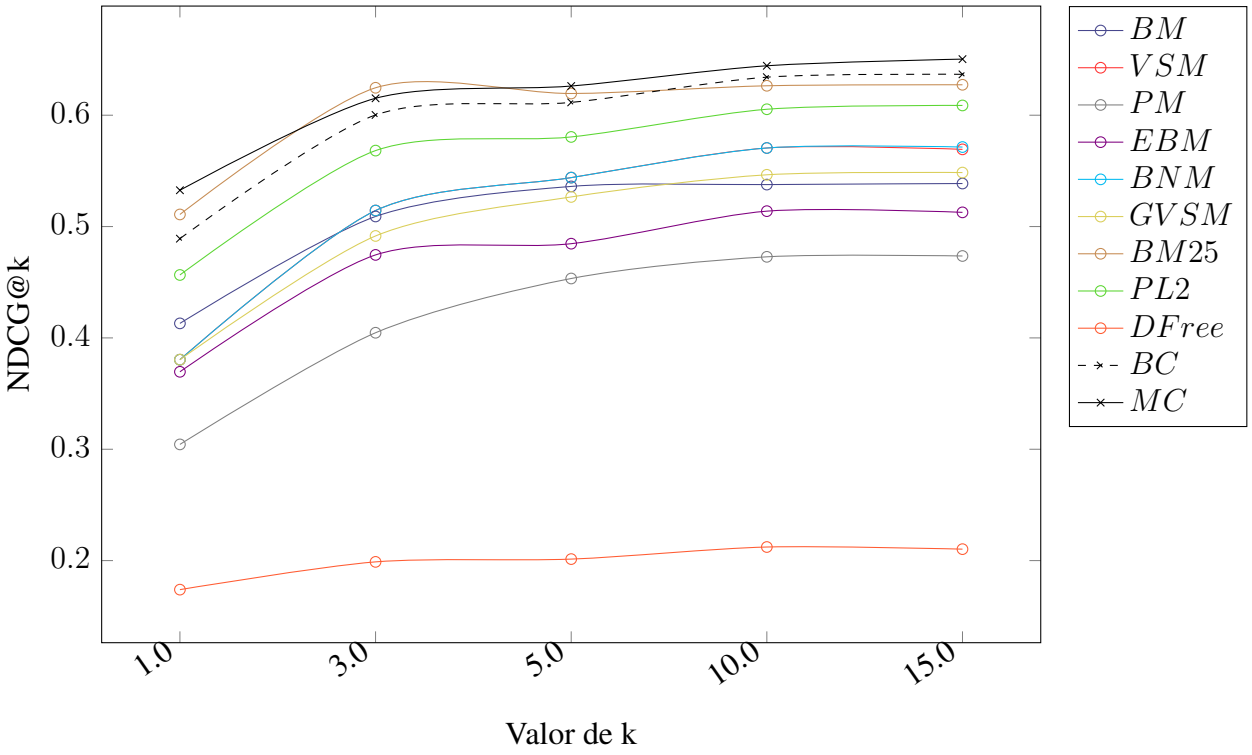

Figura 4.6 – Avaliação de NDCG@k para a coleção NPL

- novamente a métrica NDCG@k distingue melhor a eficácia encontrada pelos modelos no topo do ranqueamento, valorizando ainda mais o desempenho do modelo BM25 e das agregações BC e MC, chegando a 60% de NDCG@5;

- os modelos VSM e BNM apresentaram resultados equivalentes para todos valores de k e boa parte dos resultados relevantes retornados por suas consultas não estavam no topo do ranqueamento, dado o baixo valor de NDCG@1 e NDCG@3;
- o método de agregação MC obteve os melhores resultados no geral; desta forma, a agregação do modelo VSM ao modelo BM25, para esta coleção, melhorou os resultados de ambos os modelos considerando diferentes consultas.

Portanto, observando todos os gráficos apresentados e associados à experimentação prática do ambiente ATRI, é possível notar que os modelos probabilísticos, sobretudo o modelo BM25, apresentaram os melhores resultados. Quanto aos modelos algébricos, tem-se que o modelo VSM apresentou bons resultados para as coleções **CF** e **NPL**, mas resultados consideravelmente ruins para a coleção **LISA**. A dependência entre os termos de índice de um documento, proposta no GVSM, não melhorou o modelo VSM para as coleções avaliadas. Já os modelos booleanos BM e EBM apresentaram resultados inferiores em relação aos demais, para todas as coleções apresentadas. Por fim, as funções de ranqueamento construídas por agregações de modelos (BC e MC) apresentaram ótimos resultados em todas as coleções e comprovam que o *ensemble* de modelos de RI pode ser uma escolha adequada em diferentes cenários.

Desta forma, como uma experimentação adicional, apenas os melhores modelos (BM25, VSM, BC e MC) e o PageRank foram utilizados para avaliar um caso real de aplicação do ambiente na coleção de teste **COVID-19**. Os resultados apresentados na Figura 4.7 exibem a métrica P@k referente às consultas realizadas na coleção **COVID-19**. A partir dos resultados obtidos, nota-se que:

- os modelos comportaram-se muito bem para as consultas avaliadas, apresentando valores de P@1 de 100% e P@3 de mais de 90% para os modelos BM25 e PageRank;
- o modelo VSM apresentou uma eficácia consideravelmente inferior ao BM25, impactando negativamente os *ensembles* BC e MC, formados pela agregação com o BM25;
- o modelo PageRank obteve resultados muito similares ao BM25, indicando que a implementação do modelo utilizada segue a ordenação gerada pelo BM25, mas com ligeiras alterações.

Quanto à metrica NDCG@k para a coleção **COVID-19**, exibida na Figura 4.8, pode-se observar que:

- novamente a métrica NDCG@k é importate para distinguir a eficácia dos modelos, indicando que os modelos BM25, PageRank e MC apresentaram melhores resultados para valores inferiores de k;

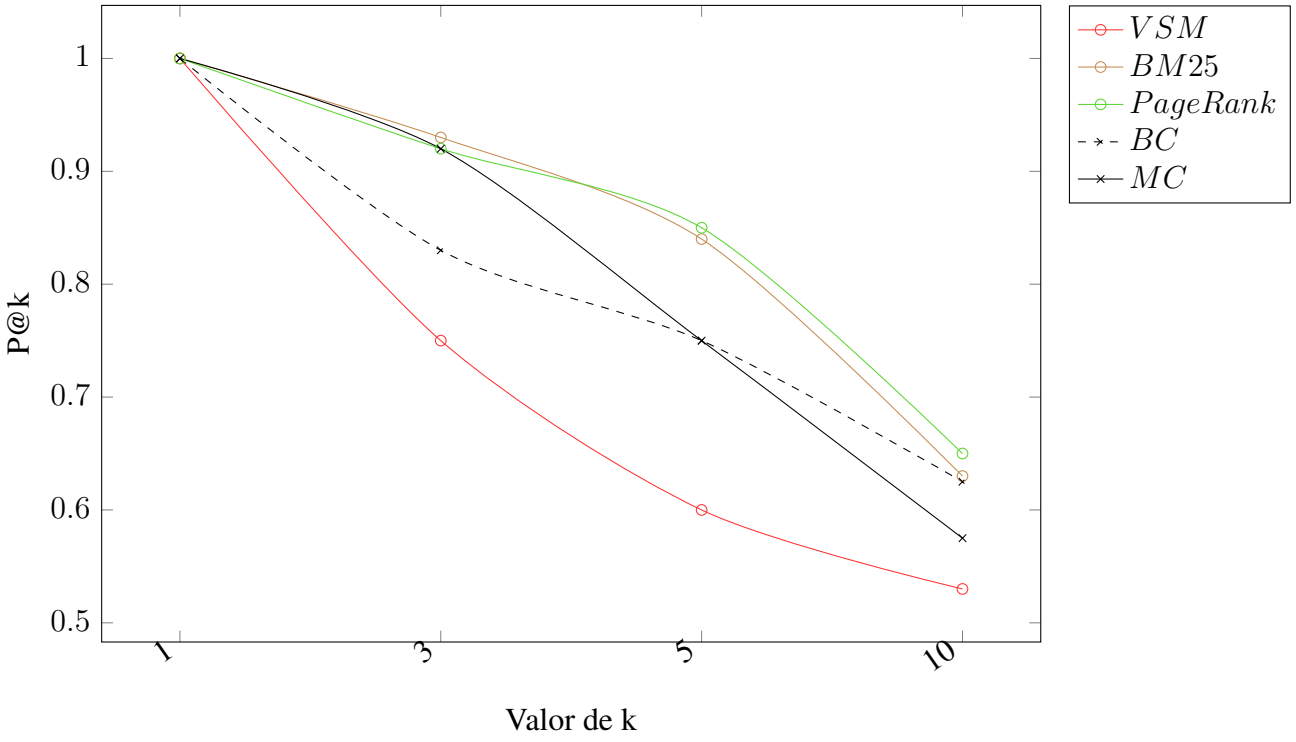

Figura 4.7 – Avaliação de P@k para a coleção COVID-19

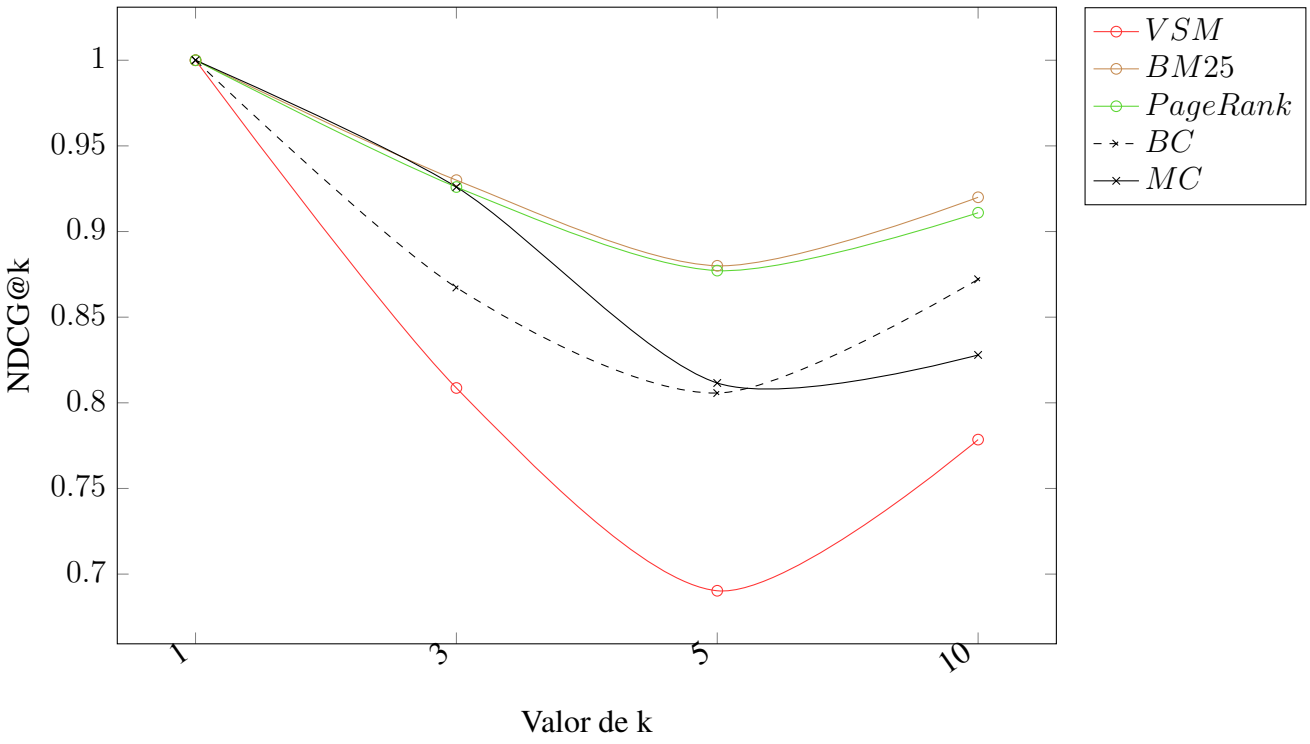

Figura 4.8 – Avaliação de NDCG@k para a coleção COVID-19

- os modelos VSM e BC apresentaram resultados valores inferiores que os demais, destacando-se negativamente o modelo VSM com 70% de NDCG@5;

De uma forma geral, o PageRank segue a ordenação gerada pelo BM25, indicando que a

ponderação, levando em conta seu grafo de autoridade de páginas, não fez grande diferença nos resultados para esta coleção em particular. Além disso, todos os modelos avaliados atingiram resultados satisfatórios, com destaque novamente para o modelo BM25 que comprova o fato dele ser um modelo amplamente utilizado como *baseline* para criação de novos modelos na literatura de RI.

5 Considerações Finais

Neste capítulo, são apresentadas as considerações finais sobre o trabalho proposto e desenvolvido, além dos testes realizados envolvendo diferentes funções de ranqueamento. A Seção 5.1 apresenta as conclusões obtidas e a Seção 5.2 as perspectivas de trabalho futuro.

5.1 Conclusão

Como já apresentado, este trabalho propõe desenvolver um ambiente de RI, denominado ATRI, para cálculo de similaridade entre consultas e coleções de documentos, possibilitando a geração de ranqueamentos de relevância para cada consulta por meio da utilização de diferentes modelos de RI.

Buscando avaliar a versão do ATRI desenvolvida, como visto, foram realizados experimentos diretamente no ambiente considerando 4 coleções distintas e, para cada coleção, os resultados de eficácia mostraram-se satisfatórios. Particularmente, conclui-se que:

- quanto aos modelos booleanos BM e EBM, foi possível observar que, de uma forma geral, obtiveram alguns bons resultados isolados, embora seus resultados de eficácia tenham sido inferiores aos demais modelos; particularmente, ao modelo EBM, um ajuste do parâmetro p pode ser necessário visando melhorar seus resultados;
- os modelos algébricos VSM, GVSM e BNM apresentaram resultados muito parecidos no geral, tendo sido bem interessantes nas coleções **CF** e **NPL**;
- os modelos probabilísticos PM e BM25 apresentaram ótimos resultados de eficácia; o modelo BM25, especificamente, pode ser considerado o melhor modelo de RI avaliado na experimentação realizada;
- os modelos PL2 e DFRee, de uma forma geral, não apresentaram resultados tão bons de eficácia, com exceção para a coleção **LISA**; destaca-se, particularmente, que o modelo PL2 apresentou resultados mais interessantes do que o DFRee para todas as coleções;
- as funções de ranqueamento baseadas em agregação com BC e MC apresentaram-se como uma alternativa muito interessante para melhoria da eficácia dos modelos de RI; em especial, tais funções melhoraram os valores do BM25 nas coleções **CF** e **NPL**.

5.2 Trabalhos Futuros

Nesta seção, são apresentadas algumas perspectivas de trabalho futuro. Desta forma, pretende-se: (1) realizar novos estudos sobre a eficácia dos modelos considerando coleções mais robustas como as da TREC; (2) desenvolver técnicas de expansão de consultas no ATRI para melhoria da eficácia das mesmas pelo ambiente; (3) desenvolver técnicas de realimentação de relevância no ATRI para melhorar a eficácia dos resultados a partir do *feedback* do usuário; e (4) desenvolver técnicas de *Learning To Rank* (LIU, 2011; LI, 2014) no ATRI para criação de *ensembles* mais robustos.

Referências

ALVAREZ, G. M.; GONÇALVES, A. L. Qualidade da informação e recuperação de informação: uma revisão da literatura. *Revista Tecnologia da Informação e Comunicação: Teoria e Prática*, v. 1, n. 1, 2017.

AMATI, G.; RIJSBERGEN, C. J. V. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)*, ACM New York, NY, USA, v. 20, n. 4, p. 357–389, 2002.

ASLAM, J. A.; MONTAGUE, M. Models for metasearch. In: *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.: s.n.], 2001. p. 276–284.

BAEZA-YATES, R.; RIBEIRO-NETO, B. *Recuperação de Informação - 2ed: Conceitos e Tecnologia das Máquinas de Busca*. Bookman Editora, 2013. ISBN 9788582600498. Disponível em: <https://books.google.com.br/books?id=YWk3AgAAQBAJ>.

BERRY, M. W.; DRMAC, Z.; JESSUP, E. R. Matrices, vector spaces, and information retrieval. *SIAM review*, SIAM, v. 41, n. 2, p. 335–362, 1999.

BIANCHINI, M.; GORI, M.; SCARSELLI, F. Inside pagerank. *ACM Transactions on Internet Technology (TOIT)*, ACM New York, NY, USA, v. 5, n. 1, p. 92–128, 2005.

BRIN, S.; PAGE, L. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, Elsevier, v. 30, n. 1-7, p. 107–117, 1998.

CAMBAZOGLU, B. B.; BAEZA-YATES, R. Scalability and efficiency challenges in large-scale web search engines. In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. [S.l.: s.n.], 2016. p. 1223–1226.

CROFT, W. B.; HARPER, D. J. Using probabilistic models of document retrieval without relevance information. *Journal of documentation*, MCB UP Ltd, 1979.

DUH, K.; KIRCHHOFF, K. Learning to rank with partially-labeled data. In: *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.: s.n.], 2008. p. 251–258.

ESPINHAÇO, B. do. *Rasgue o Céu. Intérprete: Bernardo Do Espinhaço. Compositor: Bernardo Do Espinhaço. In: THARDI*. 2018.

FERNEDA, E. *Recuperação de Informação: Análise sobre a contribuição da Ciência da Computação para a Ciência da Informação*. Tese (Doutorado) — Universidade de São Paulo, 2003.

GLEICH, D. F. Pagerank beyond the web. *siam REVIEW*, SIAM, v. 57, n. 3, p. 321–363, 2015.

GROSSMAN, D.; FRIEDER, O. *Information Retrieval: Algorithms and Heuristics*. Springer US, 2012. (The Springer International Series in Engineering and Computer Science). ISBN 9781461555391. Disponível em: <https://books.google.com.br/books?id=PlHaBwAAQBAJ>.

GUDIVADA, V. N.; RAGHAVAN, V. V.; GROSKY, W. I.; KASANAGOTTU, R. Information retrieval on the world wide web. *IEEE Internet Computing*, IEEE, v. 1, n. 5, p. 58–68, 1997.

HE, B.; OUNIS, I. Term frequency normalisation tuning for bm25 and dfr models. In: SPRINGER. *European Conference on Information Retrieval*. [S.l.], 2005. p. 200–214.

JANEIRO, d. A. Matchup: Uma ferramenta *Web* para cálculo de similaridade entre documentos. Universidade Federal de Ouro Preto, 2017.

JONES, K. S. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, MCB UP Ltd, 1972.

JÚNIOR, M. T. A.; REZENDE, M. F. P.; ASSIS, G. T. de. Development of a focused web page crawler based on genre and content. *WWW/INTERNET 2021 AND APPLIED COMPUTING*, p. 77.

KHAN, J. A. Comparative study of information retrieval models used in search engine. In: IEEE. *2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014)*. [S.l.], 2014. p. 1–5.

KRAAIJ, W. *Variations on language modeling for information retrieval*. [S.l.]: Citeseer, 2004.

LASHKARI, A. H.; MAHDAVI, F.; GHOMI, V. A boolean model in information retrieval for search engines. In: IEEE. *2009 International Conference on Information Management and Engineering*. [S.l.], 2009. p. 385–389.

LI, H. A short introduction to learning to rank. *IEICE TRANSACTIONS on Information and Systems*, The Institute of Electronics, Information and Communication Engineers, v. 94, n. 10, p. 1854–1862, 2011.

LI, H. Learning to rank for information retrieval and natural language processing. *Synthesis lectures on human language technologies*, Morgan & Claypool Publishers, v. 7, n. 3, p. 1–121, 2014.

LIN, J.; CRANE, M.; TROTMAN, A.; CALLAN, J.; CHATTOPADHYAYA, I.; FOLEY, J.; INGERSOLL, G.; MACDONALD, C.; VIGNA, S. Toward reproducible baselines: The open-source ir reproducibility challenge. In: SPRINGER. *European Conference on Information Retrieval*. [S.l.], 2016. p. 408–420.

LIU, T.-Y. Learning to rank for information retrieval. Springer Science & Business Media, 2011.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. ISBN 0521865719, 9780521865715.

MELLO, R. dos S.; DORNELES, C. F.; KADE, A.; BRAGANHOLO, V. de P.; HEUSER, C. A. Dados semi-estruturados.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas inteligentes-Fundamentos e aplicações*, Manole Ltda, v. 1, n. 1, p. 32, 2003.

MORAL, C.; ANTONIO, A. de; IMBERT, R.; RAMÍREZ, J. A survey of stemming algorithms in information retrieval. *Information Research: An International Electronic Journal*, ERIC, v. 19, n. 1, p. n1, 2014.

NORRIS, J. R. *Markov chains*. [S.l.]: Cambridge university press, 1998.

OUNIS, I.; AMATI, G.; V., P.; HE, B.; MACDONALD, C.; JOHNSON. Terrier Information Retrieval Platform. In: *Proceedings of the 27th European Conference on IR Research (ECIR 2005)*. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3408), p. 517–519. ISBN 3-540-25295-9.

PANNU, M.; JAMES, A.; BIRD, R. A comparison of information retrieval models. In: *Proceedings of the Western Canadian Conference on Computing Education*. [S.l.: s.n.], 2014. p. 1–6.

QUEPID. *Stop sidelining search*. 2021. Disponível em: <https://quepid.com/>.

RIBEIRO, B. A.; MUNTZ, R. A belief network model for ir. In: *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.: s.n.], 1996. p. 253–260.

ROBERTSON, S.; ZARAGOZA, H.; TAYLOR, M. Simple bm25 extension to multiple weighted fields. In: *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. [S.l.: s.n.], 2004. p. 42–49.

ROBERTSON, S. E.; JONES, K. S. Relevance weighting of search terms. *Journal of the American Society for Information science*, Wiley Online Library, v. 27, n. 3, p. 129–146, 1976.

ROBERTSON, S. E.; WALKER, S.; JONES, S.; HANCOCK-BEAULIEU, M. M.; GATFORD, M. et al. Okapi at trec-3. *Nist Special Publication Sp*, NATIONAL INSTIUTE OF STANDARDS & TECHNOLOGY, v. 109, p. 109, 1995.

SALTON, G.; FOX, E. A.; WU, H. Extended boolean information retrieval. *Communications of the ACM*, ACM New York, NY, USA, v. 26, n. 11, p. 1022–1036, 1983.

SALTON, G.; YANG, C.-S. *On the specification of term values in automatic indexing*. [S.l.], 1973.

SILVA, R. R. da; COTA, W. M. Criação de um arquivo invertido para a recuperação de informação em grandes volumes de texto. 2004.

SINGHAL, A.; BUCKLEY, C.; MITRA, M. Pivoted document length normalization. In: ACM NEW YORK, NY, USA. *Acm sigir forum*. [S.l.], 2017. v. 51, n. 2, p. 176–184.

SOUZA, R. R. Sistemas de recuperação de informações e mecanismos de busca na web: panorama atual e tendências. *Perspectivas em ciência da informação*, SciELO Brasil, v. 11, n. 2, p. 161–173, 2006.

SPIEGEL, M. R.; SCHILLER, J. J.; SRINIVASAN, R. A. *Probabilidade e Estatística-: Coleção Schaum*. [S.l.]: Bookman Editora, 2016.

TURTLE, H.; CROFT, W. B. Inference networks for document retrieval. In: *Proceedings of the 13th annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.: s.n.], 1989. p. 1–24.

TURTLE, H.; CROFT, W. B. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems (TOIS)*, ACM New York, NY, USA, v. 9, n. 3, p. 187–222, 1991.

TURTLE, H. R.; CROFT, W. B. A comparison of text retrieval models. *The computer journal*, The British Computer Society, v. 35, n. 3, p. 279–290, 1992.

WIVES, L. K. *Técnicas de Recuperação de Informações Com Ênfase em Informações Textuais*. Tese (Doutorado) — UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL, 1997.

WONG, S. M.; ZIARKO, W.; WONG, P. C. Generalized vector spaces model in information retrieval. In: *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.: s.n.], 1985. p. 18–25.

YANG, P.; FANG, H.; LIN, J. Anserini: Enabling the use of lucene for information retrieval research. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. [S.l.: s.n.], 2017. p. 1253–1256.