SeSGx-BT: Modelagem de Tópicos utilizando Transformers aplicada em Estudos Secundários

Demetrius Moreira Panovitch

PÓS-GRADUAÇÃO FACOM-UFMS

Data de Depósito:

Assinatura:_____

SeSGx-BT: Modelagem de Tópicos utilizando Transformers aplicada em Estudos Secundários[1]

Demetrius Moreira Panovitch

Orientador: *Prof. Dr. Bruno Magalhães Nogueira*

Dissertação apresentada à Faculdade de Computação FACOM - UFMS, para o Exame de Qualificação, como parte dos requisitos necessários para a obtenção do título de Mestre em Ciência da Computação.

UFMS - Campo Grande
Agosto/2024

[1]Trabalho Realizado com Auxílio da CAPES Proc. No: 88887.716101/2022-00

Dedicatória

Aos meus pais,
Jucilene e Valter,

À minha mãe de consideração,
Carla,

À minha namorada,
Natália,

Demetrius Moreira Panovitch.

Agradecimentos

Gostaria de agradecer à minha mãe, Jucilene, por ter me dado todo o apoio e suporte necessário durante a minha jornada na vida, e em especial, durante o mestrado. Também gostaria de agradecer ao meu falecido pai, Valter, por ter, enquanto em vida, tanto me incentivado a estudar, pois, segundo ele, meus estudos são a única coisa que nunca conseguirão tirar de mim.

Também quero expressar minha gratidão pela minha mãe de consideração, Carla, que me acolheu como filho, cuidou de mim em tempos difíceis da minha vida, e continua sempre torcendo por mim. Não fosse pela senhora, eu não estaria onde estou hoje.

Obrigado à minha namorada, Natália, por ter me acompanhado durante todo esse tempo. Seu amor, companheirismo e paciência me inspiram.

Também gostaria de agradecer à minha colega de graduação e de pesquisa, Maria Luísa, pela companhia nos momentos bons e nos momentos ruins. Obrigado por ter me ajudado a chegar onde cheguei.

Obrigado à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES pelo apoio financeiro.

Por fim, gostaria de agradecer à todos aqueles que contribuíram diretamente ou indiretamente para o desenvolvimento desta pesquisa.

Abstract

Secondary studies aggregate relevant literature to a topic to evaluate them, provide an overview, interpret them, among other purposes. However, its development has a high cost in terms of time and resources, in addition to being subject to human bias at some stages, such as the identification of primary studies. This may compromise the quality and accuracy of the review. In this work, we propose an automated approach for one of the main steps of a secondary study: formulation and refinement of search strings. The approach, called SeSGx-BT, uses a deep learning-based algorithm, called BERTopic, to perform topic modeling on a set of studies used as a Quasi-Gold Standard. The topics are used to build search strings to be applied in a hybrid search strategy, which includes database search and snowballing strategies. The results demonstrated that SeSGx-BT is capable of finding a high number of relevant studies, and a low number of irrelevant studies in hybrid search environments, resulting in a greater recall and precision, respectively, when compared to SeSGx-LDA, a similar approach that uses LDA for topic extraction. These results suggest that deep learning-based approaches can capture topics with greater semantics, minimizing human effort in the stage of primary studies identification. Based on the precision and recall values obtaineds from experiments with 10 datasets, SeSGx-BT presents itself as a promising solution for automating the formulation and refinement of search strings for secondary studies, obtaining an increase of 270% in precision at most, and 20% on recall at most.

Resumo

Estudos secundários agregam literatura relevante à algum tema para avaliá-los, fornecer uma visão geral, interpretá-los, entre outros fins. No entanto, seu desenvolvimento tem um custo elevado em termos de tempo e recurso, além de estar sujeito ao viés do pesquisador em algumas etapas, como na identificação de estudos primários. Isso pode comprometer a qualidade e acurácia da revisão. Neste trabalho, é proposta uma abordagem automatizada para uma das etapas principais de um estudo secundário: formulação e refinamento de *strings* de busca. A abordagem, chamada SeSGx-BT, utiliza de um algoritmo baseado em aprendizado profundo, chamado BERTopic, para modelagem de tópicos em um conjunto de estudos utilizado como um Quasi-Gold Standard. Os tópicos são utilizados para construir *strings* de busca para serem aplicadas em uma estratégia de busca híbrida, que inclui as estratégias de busca em bases e *snowballing*. Os resultados mostraram que a SeSGx-BT é capaz de encontrar um alto número de estudos relevantes, e um baixo número de estudos irrelevantes em ambientes de busca híbrida, resultando numa maior revocação e precisão, respectivamente, quando comparada à SeSGx-LDA, uma abordagem similar que utiliza o LDA para extração de tópicos. Esses resultados sugerem que abordagens baseadas em aprendizado profundo podem capturar tópicos com maior semântica, minimizando o esforço humano na etapa de identificação de estudos primários. Com base nas métricas de precisão e revocação obtidas a partir de experimentos executados com 10 bases de dados, a SeSGx-BT se apresenta como uma solução promissora para a automação da formulação e refinamento de *strings* de busca para estudos secundários, obtendo um aumento de até 270% na precisão, e de até 20% na revocação.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

GS *Gold Standard*

SB *Snowballing*

BSB *Backward Snowballing*

FSB *Forward Snowballing*

QGS *Quasi-Gold Standard*

LDA *Latent Dirichlet Allocation*

PDF *Portable Document Format*

API *Application Programming Interface*

CBOW *Continuous Bag-Of-Words*

TF-IDF *Term Frequency-Inverse Document Frequency*

c-TF-IDF *Class-Based TF-IDF*

STS *Start Set*

RSL Revisão Sistemática da Literatura

MSL Mapeamento Sistemático da Literatura

SeSG *Search String Generator*

BERT *Bidirectional Encoder Representations from Transformers*

SeSGx *Search String Generator eXtended*

SeSGx-BT *Search String Generator eXtended* com BERTopic

SeSGx-LDA *Search String Generator eXtended* com LDA

Rel Estudos relevantes

CAPÍTULO

1

Introdução

Devido à grande quantidade de estudos sendo desenvolvidos e publicados nas mais diferentes áreas, realizar análises e definir o estado da arte atual é uma tarefa difícil. Com isso, estudos secundários passaram a ser utilizadas para reunir e agregar estudos primários, além de serem utilizadas para manter-se atualizado sobre o estado da arte de uma determinada área. Além disso, Revisões Sistemáticas da Literatura (RSLs), que são um tipo de estudo secundário, facilitam a identificação de lacunas em algum tópico, possibilitando e guiando novos estudos (Dybå and Dingsøyr, 2008). Estudos primários são estudos experimentais e estudos empíricos que medem diretamente os objetos de interesse, seja por meio de surveys, experimentos, estudos de caso, ou outro método (Kitchenham et al., 2015).

Conforme mencionado anteriormente, estudos secundários agregam e analisam estudos primários. Para isso, é necessário utilizar uma estratégia de busca que encontre os estudos necessários para a análise. Existem diferentes estratégias de busca que podem ser utilizadas, tal como busca manual, busca automatizada, e *snowballing*. Também é comum a combinação de diferentes estratégias de busca, como a proposta por Mourão et al. (2020), que consiste na busca automatizada seguida de *snowballing*, visando reduzir o esforço do pesquisador na etapa de identificação de estudos.

Na busca automatizada, é necessária a formulação de uma *string* de busca. Uma *string* de busca consiste na combinação de palavras-chave e operadores booleanos, como “AND” e “OR”, de forma a encontrar a maior quantidade de estudos que sejam relevantes para as questões de pesquisa definidas. A formulação de uma *string* de busca eficiente é uma tarefa que consome muito tempo (Kuhrmann et al. (2017), Imtiaz et al. (2013)), além de ser uma etapa

que ainda é executada de forma manual por muitos pesquisadores. O refinamento da string de busca consiste em inserir, remover, ou alterar os termos e os operadores booleanos utilizados na string de busca, visando encontrar um maior número de estudos relevantes. Esse refinamento pode ser realizado utilizando um Quasi-Gold Standard (QGS), que consiste em um conjunto de estudos relevantes já conhecidos pelo pesquisador.

Uma *string* de busca deve, idealmente, encontrar a maior quantidade de estudos relevantes possível. No entanto, além disso, também é desejável que poucos estudos não relevantes sejam encontrados, de modo a minimizar a quantidade de estudos a serem avaliados pelos pesquisadores utilizando os critérios de inclusão e exclusão. No entanto, a definição de *strings* de busca pode ser uma tarefa que requer tempo e esforço elevados devido à dificuldade em encontrar sinônimos e palavras chave para a *string* de busca, ou até mesmo a necessidade de um conhecimento elevado do domínio (Riaz et al., 2010). Com isso, é possível afirmar que a *string* de busca impacta diretamente no resultado final de um estudo secundário.

Visando automatizar a etapa de geração e refinamento de *strings* de busca, Alves et al. (2022) desenvolveram uma ferramenta baseada em mineração de textos, chamada SeSG (*Search String Generator*). Dado um conjunto de metadados de estudos, a ferramenta extrai termos relevantes e representativos desse conjunto, e os enriquece. De forma geral, a ferramenta é composta de quatro etapas: (i) pré-processamento, extração de tópicos e enriquecimento, formulação da *string* de busca, e a utilização da *string* de busca.

Outras abordagens como a de Scells et al. (2019) e de Mergel et al. (2015) também auxiliam na automatização de estudos secundários. No entanto, ambas essas abordagens não atuam na geração de *strings* de busca, realizando somente a etapa de refinamento. Isso mostra que ainda existe margem para mais propostas de automatização de estudos secundários.

Os resultados obtidos por Alves et al. (2022) na automatização de geração e refinamento de *strings* de busca fornecem à comunidade científica uma ferramenta para a rápida análise de áreas de conhecimento com uma diminuição do esforço demandado. Entretanto, o processo adotado pela SeSG apresenta alguns pontos que podem ser explorados e melhorados.

O algoritmo utilizado pelos autores para extração de tópicos é o *Latent Dirichlent Allocation* (LDA) (Blei et al., 2003), um algoritmo não supervisionado baseado em distribuições estatísticas. Apesar do LDA ser um algoritmo simples e com resultados de fácil interpretação, seu uso na extração de tópicos para a automação de *strings* de busca traz algumas limitações. O LDA é um método não supervisionado que não incorpora informações semânticas das palavras, o que pode levar a resultados imprecisos, especialmente em campos

altamente especializados.

Além disso, a determinação do número adequado de tópicos é desafiadora e sensível a parâmetros. Por fim, a suposição de *bag-of-words* do LDA não leva em conta a ordem das palavras ao gerar as representações dos termos. O mecanismo de *bag-of-words* leva em conta somente a ordem dos termos para gerar suas representações, desprezando ordem, contexto, e qualquer outro tipo de relação que possa existir entre termos.

Além disso, o LDA funciona melhor com grandes conjuntos de dados, e em situações de dados limitados, como em estudos secundários, onde o número de estudos a serem analisados não é grande, os resultados podem ser menos confiáveis. Outra característica fundamental deste método é sua inicialização aleatória, o que pode tornar os resultados inconsistentes.

A proposta desse trabalho consiste no uso de outro algoritmo de extração de tópicos, chamado BERTopic (Grootendorst, 2022), que utiliza o BERT para gerar as representações dos termos. O BERT é um modelo de representação de linguagem, treinado em grandes bases de texto. Sua arquitetura permite uma melhor extração de representação de termos, de forma a carregar maior significado semântico, levando em conta o contexto bidirecional de onde os termos estão inseridos. Além disso, o BERTopic não possui inicialização aleatória, o que torna os resultados mais consistentes devido a ausência de qualquer vantagem ou desvantagem proveniente de uma inicialização aleatória. Com isso, é esperado que a utilização desse algoritmo resulte em *strings* de busca com uma melhor performance em termos de precisão e revocação, ou seja, que encontre poucos estudos irrelevantes, e a maioria dos estudos relevantes, respectivamente.

Para apoiar a execução dos experimentos, desenvolvemos uma ferramenta chamada SeSGx[1], que é baseada no processo proposto por Alves et al. (2022). Essa ferramenta nos permite gerar as *strings* de busca utilizando diferentes estratégias para cada uma das etapas do processo.

Dessa forma, este trabalho propõe a SeSGx-BT, uma instância da SeSGx que utiliza o BERTopic para extração de tópicos ao invés do LDA. Visando comparar a performance dos dois algoritmos de extração de tópicos na ferramenta, o experimento realizado por Alves et al. (2022) foi replicado utilizando a SeSGx-BT e a SeSGx-LDA, também uma instância da SeSGx utilizada nesse projeto, que usa o LDA para extração de tópicos.

No entanto, para obter uma maior generalização dos resultados em comparação aos obtidos por Alves et al. (2022), foram utilizados 10 estudos secundários para a execução dos experimentos, incluindo os três utilizados por Alves et al. (2022). Além disso, ao utilizar um maior número de estudos para os

[1] https://github.com/sesgx

experimentos, conseguimos avaliar o comportamento da abordagem em ambientes diferentes, devido a variações no tamanho de cada estudo, ou seja, na quantidade de estudos agregados pelo estudo secundário. Também há variação nos temas dos estudos secundários utilizados.

Os resultados da SeSGx-BT mostraram um aumento de até 270% na precisão, com uma precisão média de 0.11 e de até 20% na revocação, com uma revocação média de 0.73 quando comparados à SeSGx-LDA. Além disso, a SeSGx é mais modular em relação à SeSG, possibilitando que outros algoritmos sejam testados e utilizados em cada etapa da ferramenta. Com a SeSGx também obtivemos uma redução considerável no tempo de execução, reduzindo de aproximadamente 24 horas de execução dos experimentos para 2 horas.

1.1 Objetivos e Hipótese

O objetivo deste trabalho é verificar se a utilização de modelos de linguagem, como o BERT, na ferramenta proposta por Alves et al. (2022), especificamente na etapa de extração de tópicos da SeSGx, fornece melhores resultados em termos de redução de esforço ou maior completude em um estudo secundário, quando comparados a utilização do LDA.

A hipótese definida para este trabalho é que a utilização de um método de extração de tópicos baseado em um modelo de linguagem irá apresentar desempenho superior, seja em termos de revocação, precisão, ou ambos, devido à sua capacidade de utilizar representações de termos que levam em conta o contexto no qual se está inserido, ao invés de uma representação estatística. Para avaliar a validade da hipótese, definimos as seguintes questões de pesquisa:

1. O BERTopic é capaz de gerar tópicos que consigam desempenho melhor que o LDA em termos de precisão e revocação na geração automática de *strings* de busca?

2. Métodos baseados em aprendizado profundo para a geração de tópicos, como o BERTopic, geram resultados com maior consistência na geração de *strings* de busca - isto é, menor variância em diferentes bases de dados?

3. Na geração de *strings* de busca, quais parâmetros possuem maior influência nos resultados obtidos?

1.2 Contribuições deste trabalho

De modo geral, as contribuições deste trabalho são:

- A SeSGx-BT, uma abordagem que usa um algoritmo de extração de tópicos baseado em aprendizado profundo (BERTopic) para apoiar a automação de estudos secundários baseados em busca híbrida.
- Refatoração da SeSG, tornando-se a SeSGx. Essa refatoração trouxe uma maior modularidade e flexibilidade para executar o processo proposto por Alves et al. (2022).
- Mostrar que uma abordagem que utiliza um algoritmo baseado em aprendizado profundo resulta em uma redução do esforço do pesquisador na etapa de identificação de estudos, quando comparada à abordagem baseada em métodos estatísticos (LDA).
- Uma extensão dos resultados obtidos por Alves et al. (2022), visto que utilizamos 10 bases de dados, onde sete são novos.
- Ablação dos parâmetros de formulação da *string*, visando entender o impacto de cada parâmetro na performance da *string*.

1.3 Organização do Texto

O restante deste trabalho está dividido como segue. No Capítulo 2, iremos definir estudos secundários, assim como o seu processo de desenvolvimento, detalhando conceitos como *strings* de busca, *snowballing*, e métricas de completude. No Capítulo 3, definimos processamento de textos e seu papel na automação de estudos secundários. No Capítulo 4, apresentamos a proposta desse trabalho, a SeSGx-BT, explicando o processo proposto para a abordagem. No Capítulo 5, explicamos o procedimento experimental e os resultados obtidos. Por fim, no Capítulo 6, temos a conclusão do trabalho, mostrando as diferenças entre as abordagens, e as respostas às questões de pesquisa.

CAPÍTULO

2

Estudos Secundários

Estudos secundários são realizados com o objetivo de revisar estudos primários relativos a uma questão de pesquisa específica, com o objetivo de integrar e sintetizar a evidência coletada a respeito dessa questão de pesquisa (Kitchenham et al. (2015), Kuhrmann et al. (2017)).

São identificados alguns tipos estudos secundários, tal como Revisões Sistemáticas da Literatura (RSL), e Mapeamentos Sistemáticos da Literatura (MSL). Kitchenham et al. (2015) definem RSLs como um estudo que identifica, avalia, e interpreta os estudos relevantes à algum fenômeno específico. Já MSLs têm por objetivo identificar e classificar outros estudos acerca de um tópico.

Para identificar a etapa do processo de estudos secundários na qual a SeSGx é aplicada, iremos utilizar as diretrizes fornecidas por Kitchenham et al. (2015) para a condução de Revisões Sistemáticas da Literatura. No entanto, é importante notar que a SeSGx pode ser utilizada em qualquer estudo secundário que precise de uma *string* de busca.

Conforme demonstrado na figura 2.1, uma Revisão Sistemática da Literatura é dividida em três fases: Planejamento, Execução, e Documentação. Este trabalho visa atuar na fase de Execução, mais especificamente na atividade de Identificação de Estudos. Kitchenham (2004) define o objetivo dessa atividade como sendo a identificação de estudos relevantes para o tema desejado, utilizando as estratégias de busca definidas previamente.

2.1 Estratégias de busca

Como já mencionado, um passo essencial do processo de estudos secundários é a definição de uma estratégia de busca. Idealmente, a estratégia defi-

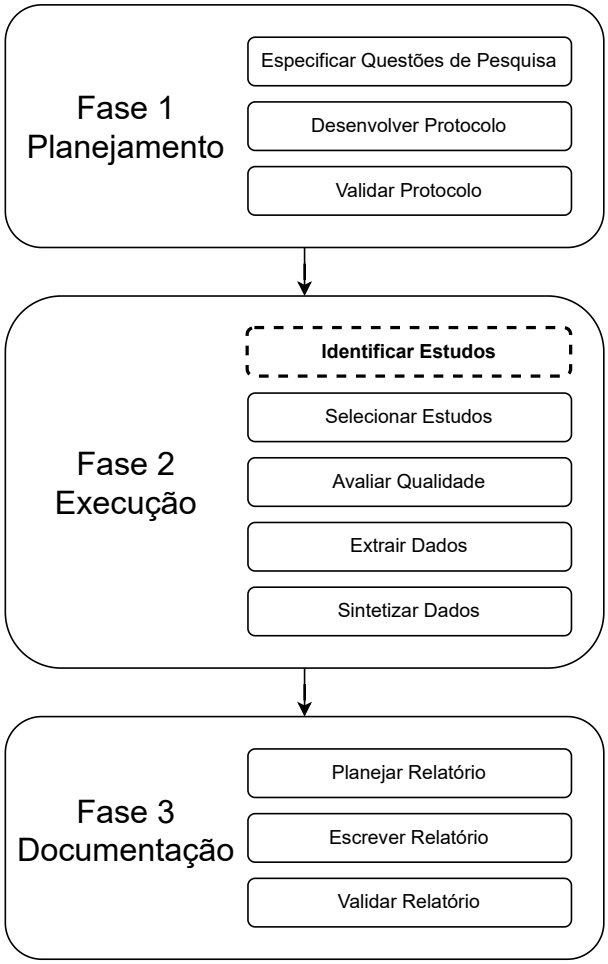

Figura 2.1: Fases do Processo de uma Revisão Sistemática da Literatura. Baseado em (Kitchenham et al., 2015).

nida deve encontrar o máximo de estudos relevantes possível para responder à questão de pesquisa.

Entre diversas estratégias disponíveis, duas das mais recorrentes são busca automatizada e *snowballing*. É importante notar que a estratégia de *snowballing* tem uma melhor performance quando utilizada em conjunto com a busca automatizada (Mourão et al., 2020), se tornando assim uma estratégia de busca híbrida. Com isso, a estratégia de busca automatizada não deve ser substituída pela estratégia de *snowballing*.

2.1.1 Busca automatizada

A estratégia de busca automatizada utilizando *strings* de busca envolve o uso de recursos digitais (e.g., bases de dados, bibliotecas digitais, sistemas de indexação) para encontrar estudos relevantes. É importante que o pesquisador documente os recursos escolhidos, além de especificar a *string* de busca utilizada.

Strings de busca são definidas e refinadas de forma iterativa. Geralmente, as palavras-chave utilizadas na *string* de busca são derivadas das questões de

pesquisa. No entanto, também podem ser utilizados alguns termos recorrentes dos estudos relevantes já encontrados (artigos de controle). Uma técnica que pode ser utilizada para aumentar o alcance da busca é adicionar sinônimos das palavras-chave na *string* de busca (Kitchenham et al., 2015).

A combinação lógica desses termos, utilizando operadores booleanos (e.g., "AND" e "OR") deve ter em vista um balanceamento entre uma busca que retorne o máximo de estudos relevantes (i.e., alta revocação) e que retorne o mínimo de estudos irrelevantes (i.e., alta precisão) (Kitchenham et al. (2015)).

É natural que diferentes áreas de conhecimento utilizem os mesmos termos, porém com significados diferentes. Isso faz com que as buscas em bases de dados retornem estudos que não necessariamente fazem parte do contexto da pesquisa. Uma estratégia eficiente para manter-se próximo ao contexto da busca é utilizar *snowballing*, visto que há uma minimização do ruído proveniente de outras áreas de conhecimento (Badampudi et al. (2015)).

2.1.2 Snowballing

A estratégia de *snowballing* consiste na análise das citações dos estudos relevantes encontrados até o momento. Dado o contexto onde as referências e citações estão introduzidas, é mais fácil obter informação sobre o conteúdo do estudo quando comparado a tentar encontrar o estudo via busca automatizada (Wohlin (2014)).

Segundo Badampudi et al. (2015), a eficiência do *snowballing* é comparável a eficiência da busca automatizada, podendo, as vezes, obter resultados de completude melhores. No entanto, essa eficiência elevada está diretamente ligada a construção de um *start set* adequado. O *start set* é o conjunto inicial de estudos que serão utilizados como ponto de partida para o *snowballing*. Portanto, para uma alta eficiência do *snowballing*, é esperado que o *start set* tenha conexões (através de citações) com todos os estudos relevantes do *Gold Standard* (GS). O GS representa o conjunto de estudos que definitivamente devem fazer parte do estudo secundário. O conceito de GS é elucidado com mais detalhes na Seção 2.3.

Neste sentido, é aceitável afirmar que uma estratégia de busca híbrida combinando busca automatizada e *snowballing* irá atingir um nível aceitável de completude, dados os recursos humanos e o tempo disponível para a execução do estudo (Mourão et al., 2020), podendo ser mais efetiva do que utilizar somente a busca automatiza (Wohlin et al., 2022). Na próxima seção, serão discutidas formas de avaliar a completude de um processo de busca.

2.2 Avaliação da completude

No contexto de Teoria de Recuperação de Informação, Van Rijsbergen (1979) afirma que a efetividade da recuperação é medida usando principalmente precisão, revocação, ou alguma outra medida que as manipule. Em estudos secundários, precisão e revocação podem ser definidas da seguinte forma:

$$P = \frac{Rel_{encontrado}}{N_{total}} \tag{2.1}$$

$$R = \frac{Rel_{encontrado}}{Rel_{total}} \tag{2.2}$$

P significa precisão, R significa revocação, $Rel_{encontrado}$ é o número de estudos relevantes encontrados pela busca, Rel_{total} é o número total de estudos relevantes existentes e N_{total} é o número total de estudos encontrados pela busca.

Como já mencionado, uma busca ótima deve possuir uma alta revocação, indicando que grande parte dos estudos relevantes foram encontrados. No entanto, maximizar a revocação invariavelmente reduz a sua medida complementar, que é a precisão.

A precisão deve ser levada em conta, visto que uma alta precisão indica que o número de estudos irrelevantes que devem ser analisados é reduzido, diminuindo a carga de trabalho dos pesquisadores. Tendo isso em mente, é necessário balancear entre uma boa revocação, e uma boa precisão (Dieste et al. (2009)).

O $F\text{-}Score$, derivado da "medida de efetividade" proposta por Van Rijsbergen (1979), é definido como a média harmônica entre precisão e revocação. Em sua forma geral, é necessário um parâmetro β, que indica em quantas vezes a revocação é mais importante que a precisão. Um $F\text{-}Score$ balanceado (i.e., $\beta = 1$) significa que tanto a precisão quanto a revocação possuem a mesma importância.

A fórmula geral do $F\text{-}Score$ é definida da seguinte forma:

$$F_\beta\text{-}Score = (1+\beta^2)\frac{P \times R}{(\beta^2 \times P) + R} \tag{2.3}$$

Já a fórmula para um $F\text{-}Score$ balanceado é definida da seguinte forma:

$$F_1\text{-}Score = \frac{2 \times P \times R}{P + R} \tag{2.4}$$

Durante a execução de um estudo secundário, sem domínio prévio da área, não se sabe o número total de estudos relevantes. Com isso, é impossível calcular as métricas de completude definidas anteriormente. No entanto, é

possível avaliar a completude utilizando um subconjunto de estudos que são relevantes. Um exemplo desse caso de uso é o *Quasi-Gold Standard* (QGS), que será discutido na próxima seção.

2.3 Avaliação da qualidade

Conforme mencionado anteriormente, o *Gold Standard* representa o conjunto de estudos relevantes para as questões de pesquisa e que devem fazer parte do corpo de conhecimento do estudo secundário (Zwakman et al., 2018). Em áreas como medicina e ciências sociais, o conceito de GS tem sido utilizado para melhorar a busca de literatura em RSLs (Dickersin et al. (1994), White et al. (2001)).

Segundo Zhang et al. (2011), grande parte das RSLs em engenharia de software não possui um GS. Para resolver esse problema, Zhang et al. (2011) introduz o conceito de *Quasi-Gold Standard*, um conjunto de estudos conhecidos sobre um determinado tópico de pesquisa.

Utilizando o conceito de QGS, Kitchenham et al. (2015) propõe dois usos para este conjunto de estudos (também ilustrados na figura 2.2):

1. Refinar a *string* de busca
2. Avaliar a qualidade da busca através da revocação

Adaptando para o nosso contexto, utilizaremos as seguintes métricas para avaliar uma *string* de busca:

- $\text{Precisão}_{\text{sts}}$: A precisão do *start set* baseado nos estudos retornados pela base.

$$\text{Precisão}_{\text{sts}} = \frac{N_{gs\_scopus}}{N_{total}} \tag{2.5}$$

- $\text{Revocação}_{\text{sts}}$: A revocação do *start set* baseado nos estudos retornados pela base.

$$\text{Revocação}_{\text{sts}} = \frac{N_{gs\_scopus}}{GS_{tamanho}} \tag{2.6}$$

- F1_{sts}: O F_1-score do *start set*.

$$\text{F1}_{\text{sts}} = \frac{2 \cdot \text{Precisão}_{\text{sts}} \cdot \text{Revocação}_{\text{sts}}}{\text{Precisão}_{\text{sts}} + \text{Revocação}_{\text{sts}}} \tag{2.7}$$

- $\text{Revocação}_{\text{bsb}}$: A revocação considerando a utilização de *backward snowballing*.

$$\text{Revocação}_{\text{bsb}} = \frac{N_{gs\_scopus+bsb}}{GS_{tamanho}} \tag{2.8}$$

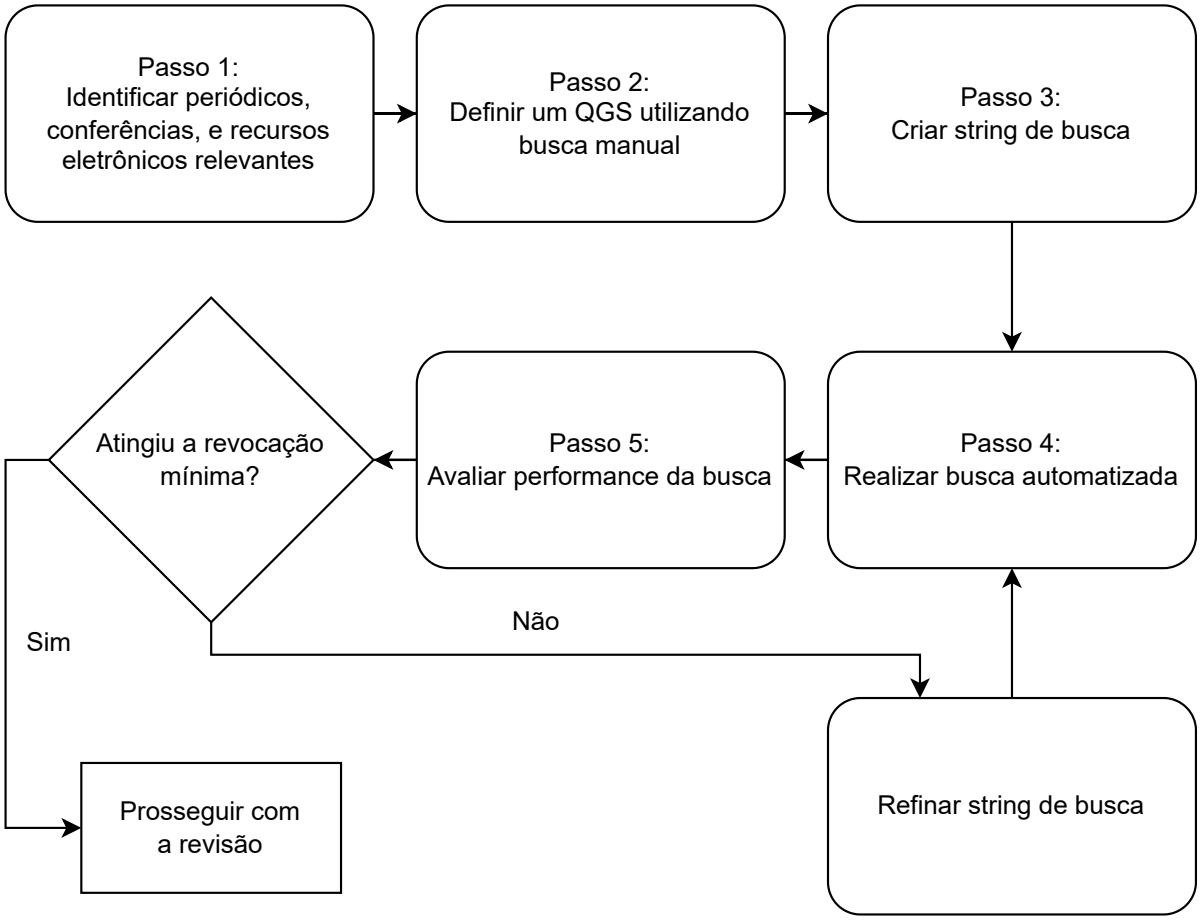

Figura 2.2: Utilização do QGS em uma RSL. Baseado em (Kitchenham et al., 2015).

- Revocação$_{\text{sb}}$: A revocação considerando a utilização de *backward* e *forward snowballing*.

$$\text{Revocação}_{\text{sb}} = \frac{N_{gs\_scopus+sb}}{GS_{tamanho}} \tag{2.9}$$

Segue abaixo o significado de cada variável das equações acima:

- N_{gs_scopus}: Número de estudos do GS encontrados pela Scopus.
- N_{total}: Número total de estudos encontrados pela Scopus.
- $GS_{tamanho}$: Número de estudos no GS.
- $N_{gs\_scopus+bsb}$: Número de estudos do GS retornados pela Scopus somado ao número de estudos que seriam encontrados ao utilizar *backward snowballing*.
- $N_{gs\_scopus+sb}$: Número de estudos do GS retornados pela Scopus somado ao número de estudos que seriam encontrados ao utilizar *backward* e *forward snowballing*.

2.4 Automação de Estudos Secundários

Devido ao aumento no número de estudos secundários, foram desenvolvidas ferramentas para auxiliar no desenvolvimento de estudos secundários. Naturalmente, também foram desenvolvidos alguns estudos com o objetivo de analisar as ferramentas disponíveis.

van Dinter et al. (2021) realizaram uma Revisão Sistemática da Literatura com o objetivo de sintetizar estudos que foquem na automação de estudos secundários. De acordo com o estudo, a etapa com maior número de abordagens para automação é a de **seleção de estudos primários**. Apesar de a segunda etapa com mais automações ser a de **identificação de estudos primários**, a grande maioria das automações se resumem a apenas refinar uma *string* de busca já existente. Isso mostra que ainda há espaço para automação na atividade de **geração** e **refinamento** de *strings* de busca.

Scells et al. (2019) definiram uma estratégia para refinamento de *strings* de busca para RSLs. Essa estratégia é baseada em uma transformação em cadeia de uma determinada *string* de busca, alterando partes da *string* como operadores booleanos (*AND* e *OR*), ou adicionando sinônimos através de *embeddings*.

Ros et al. (2017) desenvolveram uma abordagem para busca e seleção semiautomática de estudos primários para RSLs. A abordagem é baseada em aprendizado por reforço, onde o pesquisador fornece um conjunto inicial de estudos primários, e a ferramenta gera uma *string* de busca que será utilizada automaticamente para mostrar os estudos retornados de forma ranqueada. A *string* é refinada iterativamente de acordo com a inclusão de novos estudos no conjunto utilizado para gerá-la.

Adeva et al. (2014) validaram o uso de algoritmos de aprendizado de máquina para classificação de textos, com o objetivo de apoiar RSLs. A técnica validada consiste em utilizar um conjunto de metadados de estudos, como título e resumo, para treinar um algoritmo que será utilizado para classificar automaticamente se uma determinada citação deve ser incluída ou não no conjunto de estudos coletados. Os resultados foram promissores, porém com uma alta variação a depender do algoritmo utilizado.

Mergel et al. (2015) definem um método para apoiar a geração de *strings* de busca. O método consiste em utilizar mineração de textos visual para sugerir novos termos possivelmente relevantes ao usuário. Com isso, o usuário decide se quer, e como irá adicionar um novo termo relevante na *string*.

Bannach-Brown et al. (2019) estudaram o uso de algoritmos de aprendizado de máquina para reduzir o esforço durante a triagem de citações, e reduzir o erro humano nessa atividade, especificamente em RSLs relaciona-

Tabela 2.1: Comparativo entre as ferramentas analisadas

Referência	**Etapa automatizada**
Scells et al. (2019)	Refinamento de *string*
Ros et al. (2017)	Geração e refinamento de *string* e Seleção de estudos primários
Adeva et al. (2014)	Triagem de citações
Mergel et al. (2015)	Refinamento de *string*
Bannach-Brown et al. (2019)	Triagem de citações
Kontonatsios et al. (2020)	Triagem de citações
Alves et al. (2022)	Geração e refinamento de *string*

das a animais. Um conjunto de estudos foi avaliado por três revisores, e um subconjunto foi utilizado para treinar algoritmos de aprendizado de máquina. O algoritmo então, para cada estudo de um outro subconjunto de validação, retornava um valor de 0 a 1, indicando a probabilidade de inclusão do estudo.

Kontonatsios et al. (2020) utilizaram redes neurais em conjunto com SVM (*Support Vector Machine*) para automatizar o processo de triagem de citações. Dado um conjunto de citações, um revisor faz a triagem de metade delas, e então esse subconjunto é utilizado como entrada para uma rede neural que irá realizar extração de características. Após isso, o algoritmo SVM é treinado utilizando esse subconjunto, com o objetivo de avaliar a outra metade das citações.

Alves et al. (2022) desenvolveram uma estratégia para automatizar a geração e refinamento de *strings* de busca no contexto de RSLs. O usuário fornece um conjunto de metadados como título, resumo, e palavras-chave de estudos, que são utilizados em um algoritmo de extração de tópicos, além da utilização de modelos de linguagem para enriquecimento de termos. Essa é a abordagem que mais se assemelha com a proposta deste trabalho, portanto, grande parte do processo proposto para a nossa abordagem foi baseada no trabalho de Alves et al. (2022).

CAPÍTULO

3

Processamento de textos

Segundo Ebecken et al. (2003), mineração de textos é "um conjunto de técnicas e processos que descobrem conhecimento inovador nos textos". Mineração de textos é uma especialização da mineração de dados, com a diferença sendo o tipo de dado com o qual cada uma trabalha: Em mineração de texto, os dados são não estruturados, enquanto que em mineração de dados, os dados são estruturados (Weiss et al., 2010).

A utilização de mineração de textos para automação de estudos secundários têm se tornado muito comum. Sundaram and Berleant (2023) mostraram que grande parte das ferramentas para automação de Revisões Sistemáticas da Literatura focam na etapa de seleção de estudos primários. Além disso, os autores encontraram apenas uma proposta para a automação da etapa de identificação de estudos primários. Essa etapa de identificação de estudos primários pode incluir a utilização de uma *string* de busca, que por sua vez pode ser automatizada utilizando mineração de textos.

3.1 Etapas da mineração de textos

Rezende et al. (2003) descrevem o processo de mineração de textos em cinco fases: Identificação do problema, Pré-processamento, Extração de padrões, Pós-processamento, e Uso do conhecimento, conforme apresentado na Figura 3.1. Essas etapas serão explicadas nas próximas Seções.

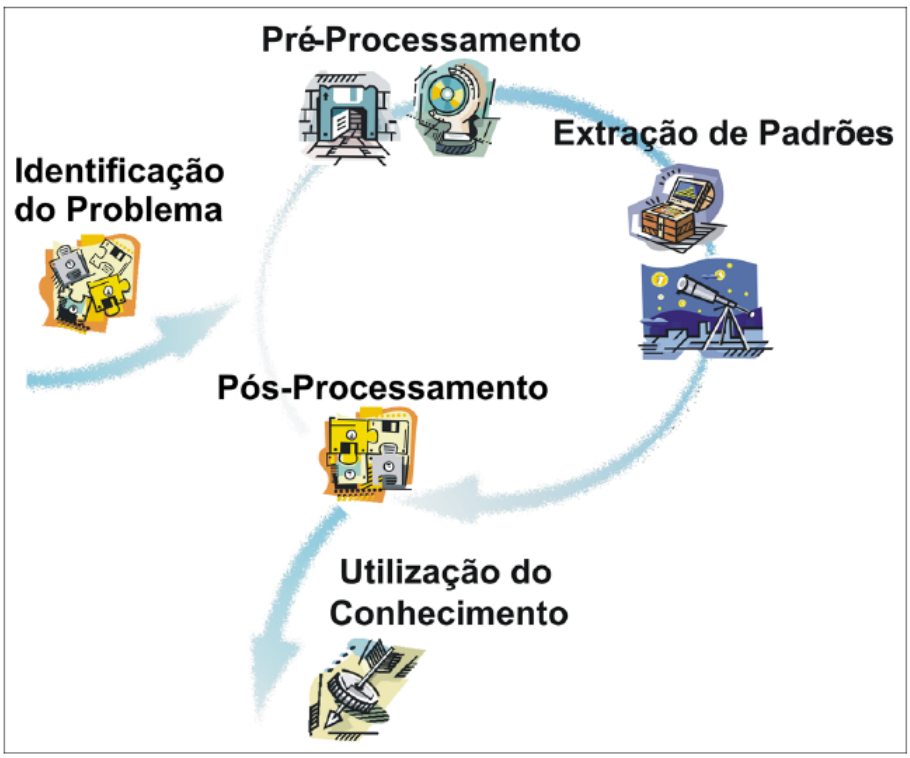

Figura 3.1: Etapas do processo de Mineração de Textos (Rezende et al., 2003).

3.1.1 Identificação do problema

Nessa fase, o escopo do problema é delimitado, e o objetivo da aplicação é definido, além das bases de textos a serem utilizadas, e da expectativa com os resultados obtidos. Rezende et al. (2003) definiram quatro questões para auxiliar nessa etapa:

- Quais são as principais metas do processo?
- Quais critérios de desempenho são importantes?
- O conhecimento extraído deve ser compreensível a seres humanos?
- Qual deve ser a relação entre simplicidade e precisão do modelo extraído?

Essas decisões irão guiar as fases seguintes. Com isso em mente, um especialista no domínio pode ser de grande ajuda nesta etapa, auxiliando na tomada dessas decisões.

3.1.2 Pré-processamento

Esta é uma das fases que mais consome tempo, visto que é nela onde os dados são ajustados para um formato adequado, utilizando atividades como tratamento, limpeza, e redução do volume de dados. Moura (2006) cita uma série de atividades que podem ser realizadas com a assistência de um especialista:

- Eliminação de repetições de documentos;
- Análise do tamanho dos documentos;
- Verificação da existência de uma estrutura nos documentos.

Após essa filtragem e tratamento inicial, os documentos devem ser transformados para uma representação processável por algoritmos de extração de padrões, tal como matrizes atributo-valor, no modelo espaço vetorial (Salton and Buckley (1987); Weiss et al. (2010)), denominado *bag-of-words*. Nesse tipo de representação, os termos são considerados independentes, o que implica que a ordem de ocorrência das palavras não importa.

Para gerar as representações de forma mais consistente, os termos são previamente analisados, utilizando, por exemplo, mecanismos para desconsiderar termos que não incorporam conhecimento útil. Essa técnica é chamada de eliminação de *stop words*. Outro objetivo dessa análise é tentar encontrar similaridades de significados entre palavras, seja reduzindo-as às suas raízes, lema, ou utilizando dicionários.

Os pioneiros na utilização de redes neurais para automatizar o aprendizado dessas representações vetoriais foram Bengio et al. (2000). Esse tipo de representação é conhecido hoje em dia como *word embeddings*, ou *embeddings*.

Word embeddings são vetores reais distribuídos em um espaço multidimensional. Além disso, cada dimensão corresponde à uma característica, podendo ser referente à semântica ou interpretação da palavra (Turian et al. (2010)). Dessa forma, a disponibilidade de um maior número de dimensões pode resultar em melhores representações, mas também no aumento do custo de processamento.

Uma das abordagens iniciais mais populares para induzir as representações foi proposta por Collobert and Weston (2008), e é baseada em uma rede neural, treinada para predizer a próxima palavra, dada uma sentença incompleta. Os pesos da rede neural são ajustados utilizando o algoritmo *backpropagation*. Dessa forma, os vetores das palavras que aparecem em contextos similares, também são similares.

Com isso, Mikolov et al. (2013) propuseram um modelo de *word embeddings* chamado Word2Vec, que popularizou o tipo de representação obtido através da estratégia descrita anteriormente. A rede neural utilizada para o Word2Vec é a mesma proposta por Collobert and Weston (2008), exceto pela ausência da camada oculta da rede, o que acelerou o processamento.

Existem duas modelagens do Word2Vec, denominadas CBOW (*Continuous Bag-Of-Words*), e Skip-gram. O CBOW consiste na predição de um palavra, dada uma janela de palavras, enquanto que o Skip-gram consite na predição

de uma janela de palavras, dada uma palavra. Ambas as modelagens estão demonstradas na Figura 3.2.

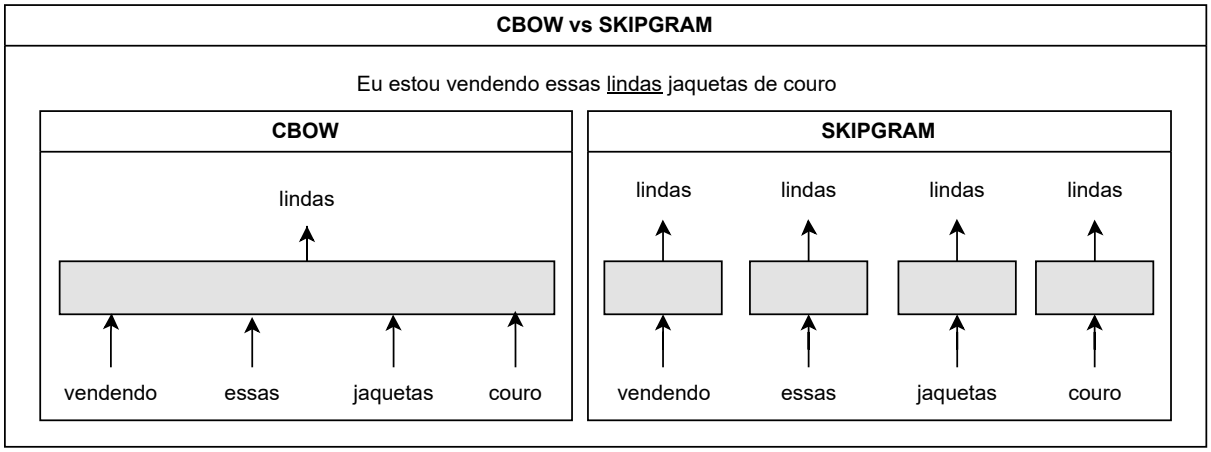

Figura 3.2: Comparativo do funcionamento do CBOW e Skip-gram. A palavra a ser predita é "lindas". Enquanto o CBOW usa uma janela de contexto para predizer a palavra, o Skip-gram usa uma palavra próxima aleatória para realizar a predição.

Um problema desses modelos apresentados é que eles são capazes de gerar representações apenas de palavras existentes no corpus. Este comportamento dificulta o uso destes modelos em aplicações cujo corpus seja diferente do que foi utilizado durante o treinamento, pois algumas palavras podem não ter vetores associados.

Para mitigar esse problema, Bojanowski et al. (2017) e Joulin et al. (2016) propuseram um modelo chamado FastText, que ao invés de utilizar janela palavras, utiliza uma janela de caracteres. Desse modo, é possível gerar vetores de palavras que não existem no corpus de treinamento, visto que somente os caracteres irão compô-la. Esse modelo é baseado no Skip-gram, do Word2Vec, com a diferença de predizer uma janela de caracteres, dado um caractere.

De forma geral, *word embeddings* são representações computacionais de termos que carregam as similaridades entre palavras. É importante notar que o tipo de similaridade (léxica, semântica, contextual, etc.) depende da estratégia utilizada para encontrá-la. Uma estratégia comum é a utilização de cosseno, onde quanto maior o cosseno do ângulo formado entre dois vetores, maior a similaridade.

Não há uma estratégia de geração de *word embeddings* que seja a melhor, sendo possível variar de acordo com o objetivos definidos durante a Identificação do Problema. Vale destacar que a etapa de Pré-processamento pode ser repetida conforme a necessidade levantada pelo Analista do processo de Mineração de Textos.

Neste trabalho, algumas das tarefas executadas durante o pré-processamento são: extração de texto de documentos PDF, extração do título, palavras-chave,

e resumo dos estudos, remoção de *stop words*, e geração das representações dos termos. Com isso, temos uma representação dos documentos manipulável por algoritmos de aprendizado de máquina. Na seção seguinte iremos elucidar como as representações podem ser utilizadas na etapa de extração de padrões.

3.1.3 Extração de padrões

As tarefas executadas nessa fase dependem do objetivo definido para a utilização do conhecimento. As atividades de extração de padrões podem ser divididas em dois grupos: preditivas e descritivas.

Atividades preditivas geram modelos para predizer o valor (rótulo) de determinadas variáveis. Já as atividades descritivas são utilizadas para produzir conhecimento a partir dos dados utilizados, por meio de padrões interpretáveis por humanos.

Com relação ao tipo de algoritmo de cada atividade, as atividades preditivas requerem algoritmos supervisionados, visto que é necessário o atributo classe do exemplo para que o algoritmo aprenda a relação entre seus atributos, e o atributo classe. Algoritmos supervisionados podem ser de classificação, ou seja, quando o atributo classe tem valor categórico, ou de regressão, quando as variáveis a serem preditas são valores reais.

Por outro lado, as atividades descritivas, utilizam algoritmos não supervisionados, que são capazes de extrair padrões de dados não rotulados. Os principais usos de algoritmos não supervisionados na área de mineração de textos são:

- Regras de associação: indicam relações entre dois conjuntos disjuntos de atributos;
- Sumarização: visa obter uma descrição compacta para um conjunto de documentos;
- Agrupamento: visa agrupar dados utilizando alguma medida de similaridade entre eles.

No contexto de mineração de textos, a atividade de sumarização mais comum é a sumarização automática de um conjunto de documentos. Isso é feito inicialmente agrupando documentos de acordo com os tópicos aos quais eles pertencem, e depois extraindo as informações mais representativas dos documentos. O objetivo é realizar essa sumarização sem perder informações essenciais do conjunto de documentos.

Neste trabalho avaliamos a utilização de algoritmos de extração de tópicos para reduzir o esforço de pesquisadores no campo de estudos secundários baseados em busca híbrida. A etapa onde a extração de tópicos pode auxiliar é na geração e refinamento de *strings* de busca, dado o grande esforço necessário para defini-la de forma a atingir uma alta completude, conforme delineado na Seção 2.

Dos diversos algoritmos disponíveis para extração de tópicos, foram avaliados: *Latent Dirichlet Allocation* (LDA), proposto por Blei et al. (2003), e BERTopic, proposto por Grootendorst (2022), ambos explicados com mais detalhes nas próximas Seções.

Neste trabalho, a extração de padrões ocorre durante a extração dos tópicos utilizando os algoritmos LDA e BERTopic, e também durante o enriquecimento dos termos usando o BERT. Para a extração de tópicos e o enriquecimento dos termos, utilizamos os metadados extraídos na etapa de pré-processamento, sendo esses: título, palavras-chave, e resumo.

3.1.4 Pós processamento e uso do conhecimento

Com os padrões obtidos, inicia-se a fase de pós-processamento, que consiste na avaliação dos aspectos do conhecimento obtido, tais como a validade, e compreensibilidade.

A validade do conhecimento é avaliada verificando se os padrões obtidos condizem com a realidade. A compreensibilidade consiste em fornecer formas de avaliar os dados, utilizando, por exemplo, mecanismos de visualização. Após a avaliação do conhecimento obtido, é possível aplicá-lo para atingir o objetivo estabelecido.

Neste trabalho, o pós processamento ocorre ao filtrar as *strings* inválidas, ou seja, que não retornaram nenhum resultado ou que não foram processadas corretamente pelo mecanismo de busca. Já o uso do conhecimento ocorre quando o pesquisador efetivamente utiliza uma das *strings* geradas em um mecanismo de busca.

3.2 Modelos de linguagem

Um ponto essencial da atividade de análise de documentos utilizando técnicas de processamento de linguagem natural é a representação computacional desses documentos. Tradicionalmente, o método utilizado para obter as representações de documentos é a *bag-of-words*. Neste método, a ordem e estrutura das palavras não são mantidas na representação, sendo mantidas apenas as palavras que ocorrem no documento, assim como a quantidade de repetições da mesma no documento.

Modelos de linguagem são ferramentas de Processamento de Linguagem Natural (PLN) que buscam entender e gerar texto humano de forma automatizada (Indurkhya and Damerau, 2010). Esses modelos são essencialmente sistemas estatísticos ou neurais que aprendem a probabilidade de ocorrência de palavras e sequências de palavras em um determinado idioma, com base em um conjunto de dados de treinamento vasto e diversificado. Eles capturam a estrutura, gramática, contexto e a semântica da língua, permitindo a realização de tarefas como tradução automática, resumo de texto, geração de texto e muito mais.

Vaswani et al. (2017) propuseram *Transformers*, uma arquitetura que utiliza mecanismos de auto-atenção para definir dependências globais entre a entrada e saída. Esta arquitetura permite maior paralelização, principalmente quando comparada à arquiteturas como Redes Neurais Recorrentes.

Mecanismos de auto-atenção são utilizados para relacionar diferentes posições de uma sequência, com o objetivo de gerar uma representação dessa sequência. Ou seja, esse mecanismo irá definir o quão importante cada posição da sequência é para a sequência como um todo. Dessa forma, as representações geradas são cientes de contexto.

No entanto, em auto-atenção, é comum que o mecanismo acabe dando muita importância para a própria palavra da qual se está gerando uma representação. Para mitigar esse problema, são utilizados blocos de Atenção Multi-Cabeça, que são análogos a utilização de múltiplos kernels em uma Rede Neural Convolucional. Ou seja, cada cabeça pode tentar capturar diferentes definições de relevância, como verbos, substantivos, entre outros.

Como os termos das sequências são processados de forma paralela, a ordem dos termos acaba sendo perdida. Para resolver esse problema, são utilizados codificadores posicionais. Estes codificadores são utilizados antes de uma *embedding* ser manipulada por uma pilha de codificadores ou decodificadores.

A arquitetura Transformers utiliza uma estrutura de codificador-decodificador, onde o codificador é responsável por transformar uma sequência de símbolos, em uma sequência de valores contínuos, e o decodificador utiliza essa sequência de valores contínuos para gerar uma sequência de símbolos para cada valor. O modelo é auto regressivo a cada passo, ou seja, utiliza as sequências anteriores como entrada para gerar a próxima.

Utilizando essa arquitetura, Devlin et al. (2019) introduziram *BERT*, um modelo de representação de linguagem pré treinado a partir de dados não anotados, com o objetivo de extrair contexto tanto antes quanto após um símbolo. Esse modelo é uma instância de Transferência de Aprendizagem, uma estratégia que consiste em adquirir conhecimento geral, para depois reutilizá-

lo em tarefas específicas.

Seu funcionamento pode ser dividido em duas etapas: Pré treinamento, e Ajuste Fino. O pré treinamento, por sua vez, também pode ser dividido em duas etapas: Modelagem de Linguagem Mascarada, e Predição de Próxima Sentença.

No inicio do pré treinamento, as *embeddings* podem ser tanto carregadas de alguma base como *Word2Vec*, como também podem ser inicializadas aleatoriamente. Com as *embeddings* carregadas, o modelo é treinado seguindo uma tarefa chamada Modelagem de Linguagem, que consiste em treinar o modelo para predizer um símbolo em uma sentença. Devido ao fato de algumas tarefas de Processamento de Linguagem Natual serem favorecidas não somente pela compreensão da relação entre símbolos, mas também pela relação entre sentenças, o modelo também é treinado seguindo uma tarefa chamada Aprendizagem de Representação. Essa tarefa consiste em treinar o modelo para predizer se uma sentença se encaixa ou não após outra sequência. Com essas duas tarefas, o modelo é capaz de aprender relações entre símbolos, e entre sequências, ambas carregando o contexto geral da sequência, e não somente do lado esquerdo, como ocorre em algumas Redes Neurais Recorrentes. Na estratégia de Transferência de Aprendizagem, a obtenção de conhecimento geral é mapeado no pré treinamento desse modelo.

Antes de utilizar, é necessário realizar ajustes no classificador do modelo, de forma que seja utilizável para a tarefa desejada. Com o classificador ajustado, basta realizar ajuste fino dos parâmetros do modelo caso necessário, treinando-o novamente utilizando os dados anotados. Algumas tarefas comuns são: análise de sentimentos, anotação de termos, entre outras.

3.3 Extração de tópicos

Modelos de extração de tópicos são ótimas ferramentas para sumarizar documentos sem a necessidade de supervisão, ou de dados anotados. O LDA é um modelo tradicional, que possui uma característica fundamental: os documentos são representados como uma *bag-of-words*, o que faz com que as representações dos termos não carreguem a relação entre palavras, ou seja, as representações não levam em conta o contexto onde o termo está inserido.

3.3.1 LDA

Idealizado por Blei et al. (2003), Latent Dirichlet Allocation (LDA) é um modelo probabilístico e generativo, utilizado para identificar tópicos de uma coleção de documentos. O método de aprendizagem desse modelo é não supervisionado, ou seja, o modelo irá identificar e extrair os padrões dos dados

sem utilizar seus rótulos.

O LDA baseia-se em dois pontos principais: (1) um documento é definido por um conjunto de tópicos, e (2) um tópico é definido por uma distribuição de probabilidade sobre os termos dos documentos (Blei et al. (2003)), conforme exemplificado na Figura 3.3. Com isso, o LDA tem por objetivo identificar os tópicos dos documentos, e, para cada tópico, estimar a distribuição de probabilidade sobre os termos.

Uma das premissas assumidas por esse algoritmo é que cada documento é apenas uma coleção de palavras (ou seja, uma *bag-of-words*). Isso faz com que fatores como a ordem das palavras, função gramatical das palavras, e o contexto no qual a palavra está inserida não são considerados.

Além disso, algumas palavras dos documentos podem ser removidas sem perda de informação, tal como artigos, preposições, conjunções, etc. Essas palavras são conhecidas como *stop words*.

Inicialmente, cada palavra de cada documento será associada à um dos tópicos aleatoriamente. Após isso, para cada palavra de cada documento, serão calculados os seguintes valores:

- Proporção de palavras do documento atual d que fazem parte de um determinado tópico t. Se a proporção é alta, é provável que a palavra atual w pertença ao tópico t. De maneira simplificada, onde $n_{d,t}$ indica quantas palavras do documento d são associadas ao tópico t, e N_d indica o número de palavras do documento d, temos a seguinte fórmula:

$$p(t \mid d) = \frac{n_{d,t}}{N_d}$$

- Proporção de documentos que contém a palavra atual w e que fazem parte de um determinado tópico t. Se a proporção é alta, é provável que o documento atual d seja descrito pelo tópico t. De maneira simplificada, onde $n_{t,w}$ indica quantas vezes a palavra w foi associada ao tópico t, e N_t indica o número de palavras associadas ao tópico t, temos a seguinte fórmula:

$$p(w \mid t) = \frac{n_{t,w}}{N_t}$$

Com essas distribuições, o algoritmo define quais termos melhor representam um tópico, e quais tópicos melhor representam um documento. As distribuições são calculadas iterativamente e de forma recorrente, até que haja convergência.

Visto que no primeiro passo do algoritmo é assumida uma distribuição aleatória, pode ser necessário executar o algoritmo outras vezes. Execuções repetidas do algoritmo podem trazer uma maior confiabilidade nos resulta-

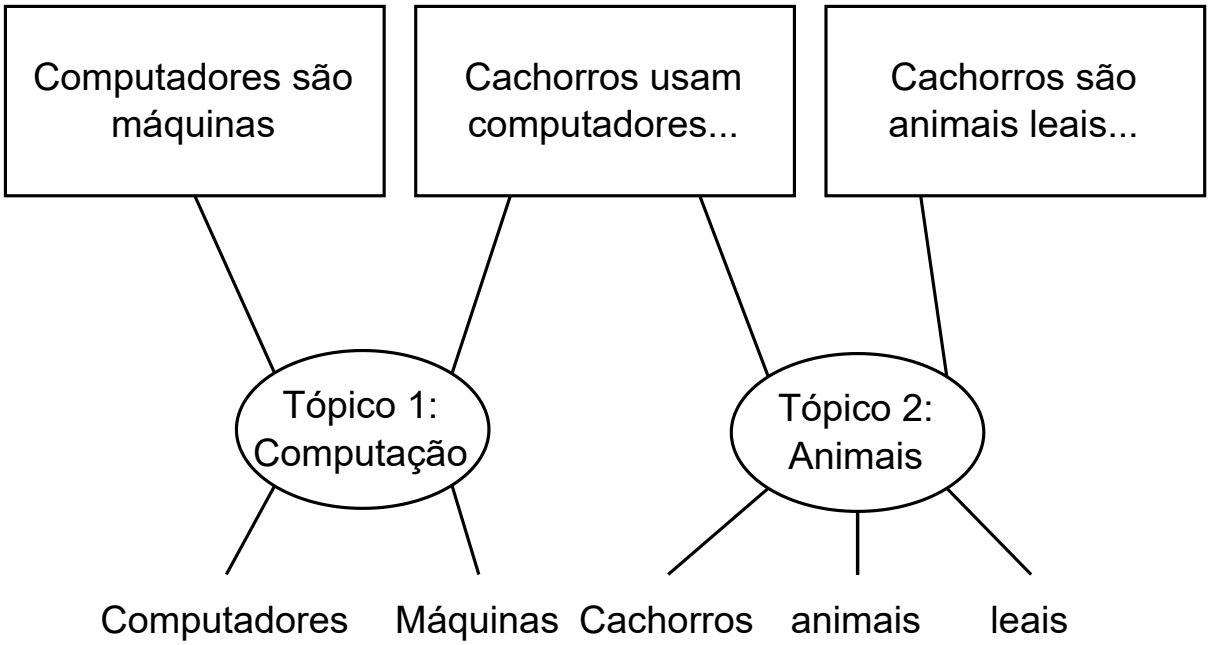

Figura 3.3: Distribuição de tópicos sobre documentos, e de palavras sobre tópicos.

dos. Ao executar várias vezes, podemos ter uma certa segurança de que os resultados não foram frutos de sorte (ou azar) na distribuição inicial aleatória.

3.3.2 BERTopic

Grootendorst (2022) desenvolveu BERTopic, um modelo que faz extração de tópicos latentes de um conjunto de documentos, utilizando modelos provenientes do BERT para gerar as *embeddings*. O grande diferencial deste modelo é a utilização de uma medida denominada *c-TF-IDF*, uma variação da medida *TF-IDF*, Frequência de Termo-Frequência Inversa de Documento.

O processo é dividido em quatro partes: Geração de *embeddings*, Redução de dimensionalidade e Agrupamento, *Tokenização*, e Extração dos tópicos, conforme demonstrado na Figura 3.4. Cada parte será descrita em seguida.

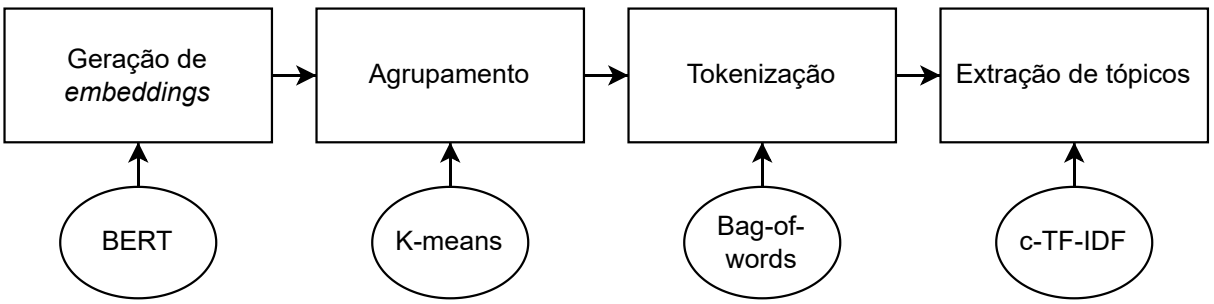

Figura 3.4: Etapas do BERTopic.

Conforme mencionado anteriormente, para gerar as *embeddings*, é utilizado um modelo que tenha como base o BERT, visto que, dessa forma, elas

irão carregar a informação do contexto em sua representação. Em seu artigo, Grootendorst utiliza o *Sentence-BERT*, que consiste em transformar sequências (frases ou parágrafos) em vetores, utilizando modelos pré-treinados. Também é possível alterar o modelo utilizado para gerar as *embeddings*, o que torna o modelo bastante modular, e podendo ser atualizado para algum modelo mais recente, ou especializado. Essas *embeddings* são utilizadas para agrupar os documentos, e não para diretamente gerar os tópicos.

Antes de realizar o agrupamento, a dimensionalidade das *embeddings* são reduzidas. Isso é necessário, pois, segundo alguns estudos (Aggarwal et al. (2001), Beyer et al. (1999)), quanto maior a dimensão dos dados, a distância entre pontos arbitrários do espaço que representam esses dados ficam cada vez mais parecidas, o que torna difícil diferenciar a distância relativa entre pares de pontos. Com as *embeddings* de dimensão reduzida, o agrupamento é realizado utilizando *HDBSCAN*, uma adaptação do algoritmo *DBSCAN* que considera hierarquia entre os grupos. No entando, por ser um algoritmo de alta densidade, para utilizá-lo é necessária uma quantidade relativamente grande de documentos. Com isso, para problemas com baixa quantidade de documentos, pode ser necessário utilizar outro algoritmo de agrupamento, como o K-means (MacQueen (1967)). No nosso trabalho, devido ao baixo número de documentos, substituímos o HDBSCAN pelo K-means na etapa de agrupamento das *embeddings*.

A *tokenização* ocorre com os documentos de um mesmo grupo sendo concatenados e utilizados em um mecanismo de *bag-of-words*. É importante notar que as representações da *bag-of-words* são obtidas por grupo ao invés de por documento, visto que estamos buscando palavras que representem um tópico, ou seja, um grupo.

A geração de tópicos é dada de tal forma que, para cada grupo seja atribuído um tópico. Para diferenciar os tópicos gerados entre grupos, é utilizada uma variação da medida TF-IDF. Essa medida, originalmente é utilizada para medir a importância de uma palavra para um documento, e foi adaptada para medir a importância de um termo para um tópico. Essa nova medida é chamada de c-TF-IDF (3.1). Para isso, todos os documentos de um grupo são concatenados e tratados como um único documento (3.5). A frequência de um termo é dada pela sua frequência em um grupo, ou seja, no documento concatenado. A frequência inversa do documento é substituída pela frequência inversa do grupo, que modela a importância de um termo para um grupo. Com isso, essa medida relaciona a importância de um termo para um grupo de documentos, ao invés de para um único documento.

$$W_{t,c} = tf_{t,c} \cdot log(1 + \frac{A}{tf_t}) \tag{3.1}$$

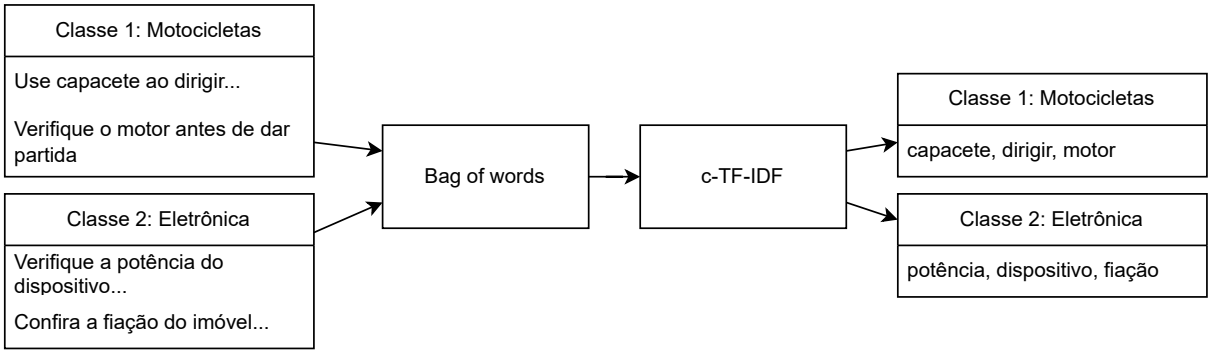

Figura 3.5: Exemplo do c-TF-IDF.

3.4 Considerações Finais

Segundo Hausner et al. (2012) e Zhang et al. (2011), é possível utilizar mineração de textos para melhorar a completude das buscas realizadas no contexto de estudos secundários. Essa melhoria pode ser fruto, por exemplo, do refinamento da *string* de busca utilizada, ou através da expansão dos termos da *string*, conforme mencionado por Thomas et al. (2011). Também conforme demonstrado por Sundaram and Berleant (2023), vemos que muitas técnicas de mineração de textos e processamento de linguagem natural vêm sendo utilizadas para automatizar diferentes etapas de estudos secundários.

Ao instanciar as etapas do processo de mineração de textos de forma adequada ao contexto de estudos secundários, é possível fornecer suporte à automação desses estudos. No contexto desse trabalho, a automação será realizada especificamente na atividade de geração e refinamento de uma *string* de busca visando alta completude e precisão para ser utilizada em bases de busca.

CAPÍTULO

4

SeSGx-BT: Search String Generator eXtended - BERTopic

Conforme mencionado anteriormente, apesar de existirem abordagens para a automação de estudos secundários, poucas delas automatizam uma etapa essencial: a geração e refinamento de *strings* de busca. Uma das abordagens que automatiza essa etapa é a SeSG, proposta por Alves et al. (2022). Essa abordagem utiliza um algoritmo de extração de tópicos baseado em distribuições estatísticas, chamado LDA. A proposta desse trabalho é fazer uma extensão da SeSG, generalizando o processo para executá-lo utilizando um algoritmo de extração de tópicos baseado em aprendizado profundo, chamado BERTopic. Para isso, desenvolvemos uma abordagem chamada SeSGx-BT.

4.1 A abordagem SeSGx-BT

O processo foi inspirado na abordagem proposta Alves et al. (2022), e é dividido nas seguintes etapas, também demonstradas na Figura 4.1:

1. **Coleta de metadados**: Consiste na extração dos metadados que serão utilizados nas próximas etapas. Os metadados a serem extraídos são: título, resumo, e palavras-chave.

2. **Extração de tópicos e enriquecimento de termos**: Utilizando os metadados coletados na etapa de pré processamento, é realizada a extração de tópicos, juntamente ao enriquecimento de termos.

3. **Formulação da *string* de busca**: A partir dos tópicos e termos enriquecidos, uma *string* de busca é formulada utilizando operadores booleanos, além da adição de limitadores de ano de publicação, caso necessários.

4. **Utilização da *string***: Consiste na utilização da *string* para realizar uma busca em bases.

5. **Avaliação do desempenho da *string***: Com os resultados retornados da busca em base, o desempenho da *string* é calculado utilizando métricas de precisão, revocação, e F_1-score.

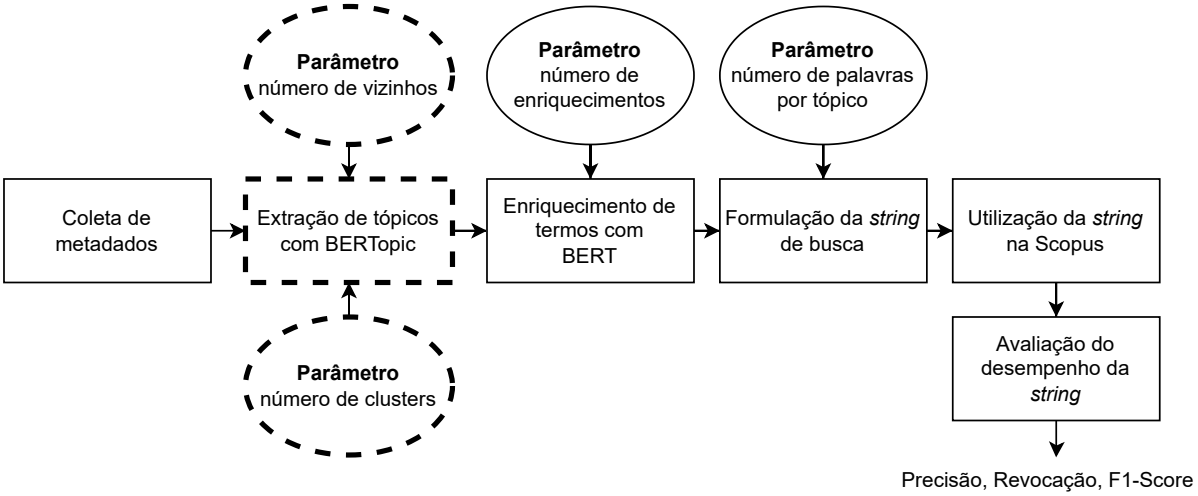

Figura 4.1: Etapas da SeSGx-BT, destacando o uso o BERTopic para a extração de tópicos.

O enriquecimento de termos pode ser feito utilizando, por exemplo, um modelo de linguagem como o BERT. O enriquecimento pode auxiliar a ampliar a busca adicionando termos similares, de modo a aumentar a revocação. Neste trabalho, o enriquecimento de um termo é feito mascarando o termo em uma sentença na qual ele apareça, e então utilizamos o BERT para predizer quais termos melhor se encaixam na máscara.

Durante o enriquecimento, é possível que ocorra repetição do próprio termo sendo enriquecido, ou que o novo termo gerado seja muito próximo do termo original, tal como singular e plural. Para mitigar isso, após o enriquecimento de um termo, os termos gerados são filtrados utilizando *stemming*, que consiste na redução de um termo para sua raíz, além da utilização da distância de Levenshtein (Levenshtein et al. (1966)), que define a quantidade de edições (inserção, remoção, substituição) para tornar uma determinada *string* em outra.

Uma *string* de busca é formulada utilizando a seguinte lógica, como demonstrado na Figura 4.2:

- Termos encontrados via enriquecimento são concatenados utilizando o operador booleano "*OR*", visto que os termos enriquecidos simbolizam sinônimos.
- Termos de um mesmo tópico são concatenados utilizando o operador booleano "*AND*", visto que o tópico completo (ou seja, seus termos utilizados em conjunto) descreve um ou mais documentos.
- Tópicos são concatenados utilizando o operador booleano "*OR*", visto que cada tópico descreve um ou mais documentos, e visamos definir uma *string* que represente todo o conjunto de documentos.

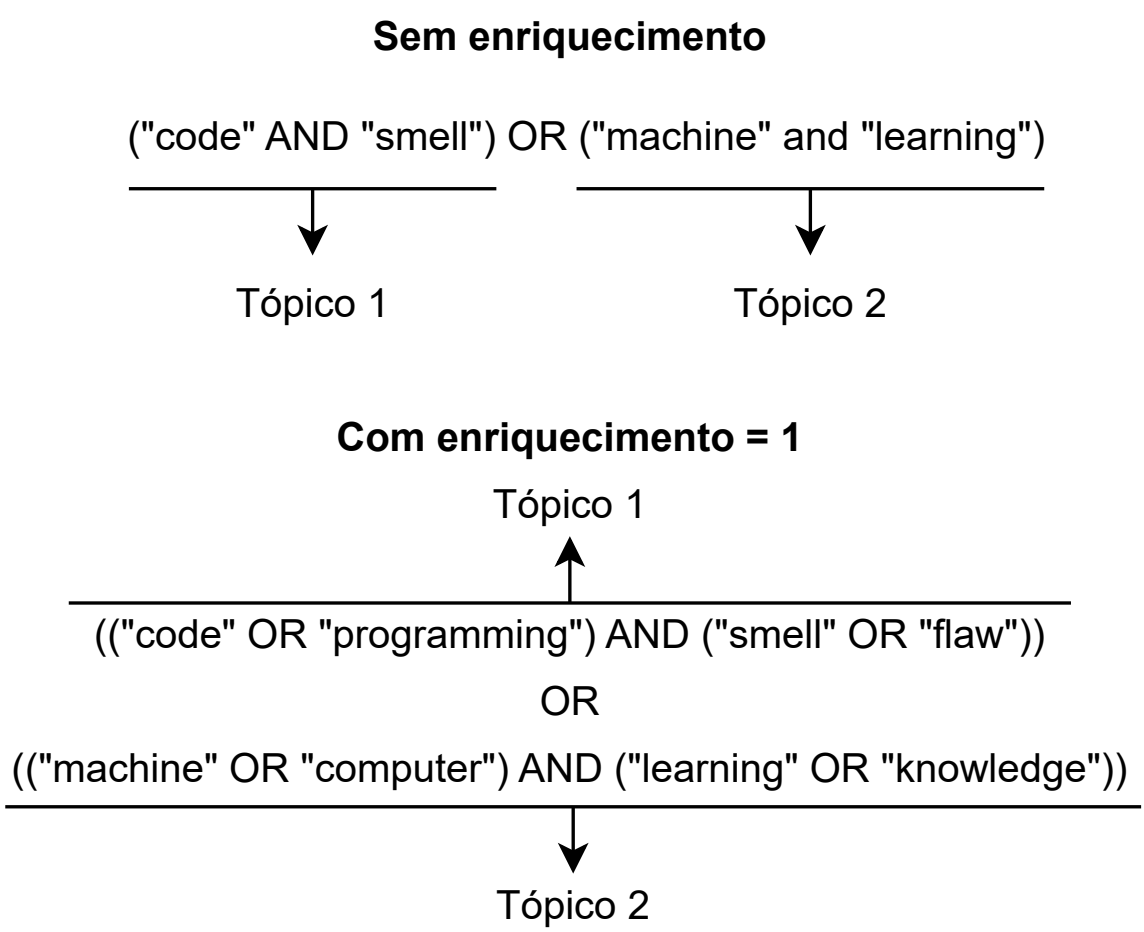

Figura 4.2: Representação de *strings* geradas por esse processo. Baseado em (Alves et al., 2022).

A avaliação do desempenho de uma *string* de busca é feita utilizando as métricas de precisão (do total de estudos retornados, quantos são relevantes), revocação (do total de estudos relevantes, quantos foram encontrados), e F_1-score (balanceamento entre precisão e revocação). A métrica de revocação é calculada considerando três situações:

- Quantos estudos relevantes foram encontrados via busca automatizada.

- Quantos estudos relevantes foram encontrados via busca automatizada em conjunto com *backward snowballing* (BSB).
- Quantos estudos relevantes foram encontrados via busca automatizada em conjunto com BSB e *forward snowballing* (FSB).

Para realizar a busca automatizada, foi utilizada a Scopus, um sistema de indexação. Já para calcular as métricas que consideram alguma técnica de *snowballing*, foi utilizado um grafo de citação gerado programaticamente a partir dos arquivos PDF correspondentes aos estudos pertencentes ao GS de cada estudo secundário.

Para executar esse processo, desenvolvemos uma ferramenta chamada SeSGx (*Search String Generator eXtended*). Além de mais modular, a SeSGx também é consideravelmente mais rápida do que a SeSG como implementada por Alves et al. (2022). Utilizando alguns componentes como banco de dados relacional, cache, programação paralela, e programação assíncrona, obtivemos um *speedup* de 12 vezes.

A SeSGx nos permite aplicar o processo proposto de forma modular e rápida. Com isso, torna-se possível a execução de mais experimentos, visando a coleta de mais dados para demonstrar a performance da ferramenta quando aplicada em estudos secundários.

A alteração principal no processo proposto é na etapa de extração de tópicos, como destacado na Figura 4.1. O algoritmo utilizado por Alves et al. (2022) para realizar extração de tópicos foi o LDA. Esse algoritmo utiliza de representações de termos baseadas em *bag-of-words*. Esse tipo de representação não carrega informações como o contexto no qual o termo está inserido, sendo representado apenas pela sua frequência no documento.

Esse algoritmo foi substituído pelo BERTopic, um algoritmo de extração de tópicos baseado em aprendizado profundo, que utiliza o BERT, um modelo de linguagem, como base. O BERTopic é capaz de extrair os tópicos utilizando representações de termos que carregam o contexto no qual o termo está inserido, o que pode fazer com que os tópicos finais sejam mais representativos sobre os documentos, o que atende à hipótese de pesquisa definida para o trabalho. Uma observação importante acerca do BERTopic, é que na etapa de agrupamento, o algoritmo *HDBSCAN*, um algoritmo de alta densidade, foi substituído pelo K-means. Isso foi necessário devido a baixa quantidade de documentos para o problema proposto.

Dessa forma, a SeSGx-BT nos ajudará a definir as vantagens e desvantagens da utilização de um método de extração de tópicos baseado em um modelo de linguagem, quando comparado à utilização do LDA. Com isso, neste trabalho teremos as seguintes instâncias da SeSGx: SeSGx-BT, que utiliza o BERTopic para extração de tópicos; e SeSGx-LDA, que utiliza o LDA para

extração de tópicos. Em ambas as instâncias, o modelo de enriquecimento utilizado foi o mesmo utilizado por Alves et al. (2022), o BERT.

4.2 Considerações finais

Neste trabalho, exploramos a automação da geração e refinamento de *strings* de busca para estudos secundários, uma etapa fundamental em que poucas abordagens existem atuam. Essas limitações nos levaram a desenvolver a ferramenta SeSGx, e a abordagem SeSGx-BT.

A SeSGx (*Search String Generator eXtended*) representa um avanço significativo em relação a SeSG, proposta por Alves et al. (2022). As melhorias implementadas na ferramenta reduzem drasticamente o tempo de execução, possibilitando um maior número de experimentos, que resulta em uma coleta de dados mais robusta, ajudando a avaliar o desempenho da ferramenta de forma mais detalhada.

A principal contribuição deste trabalho é a SeSGx-BT, uma abordagem baseada em aprendizado profundo para automação de estudos secundários. A SeSGx-BT utiliza o BERTopic ao invés do LDA, oferecendo uma representação contextualizada dos termos dos documentos, em contrapartida às representações baseadas em *bag-of-words*.

A SeSGx e a SeSGx-BT representam contribuições importantes para a automação de estudos secundários, atuando em uma etapa pouco explorada pelas abordagens existentes. Esperamos que a SeSGx facilite e aprimore futuras investigações sobre a aplicação desse processo para automação de estudos secundários.

No capítulo seguinte, avaliamos a SeSGx-BT e a SeSGx-LDA através de métricas como precisão, revocação e F_1-score, considerando diferentes cenários, incluindo buscas automatizadas e combinações com técnicas de *snowballing*. Os resultados mostram que a SeSGx-BT foi, em grande parte dos cenários, superior abordagem tradicional baseada em LDA, validando a hipótese de que métodos de extração de tópicos baseados em modelos de linguagem oferecem vantagens significativas.

CAPÍTULO

5

Experimentos e resultados

Neste capítulo serão apresentados os seguintes pontos: a ferramenta SeSGx, o funcionamento da SeSGx-BT, a estratégia para execução dos experimentos, riscos e limitações notados durante o trabalho, e os resultados obtidos.

5.1 Estratégia dos experimentos

Para avaliar o desempenho das *strings* de busca geradas pela SeSGx-BT e SeSGx-LDA, serão coletadas métricas de $\text{Precisão}_{\text{sts}}$, $\text{Revocação}_{\text{sts}}$, F1_{sts}, $\text{Revocação}_{\text{bsb}}$, $\text{Revocação}_{\text{sb}}$, e N_{total}. Além disso, também será calculada a média e o desvio padrão de cada uma das métricas citadas anteriormente.

Neste trabalho são analisados e comparados os resultados das duas estratégias a partir das médias de $\text{Precisão}_{\text{sts}}$, $\text{Revocação}_{\text{sts}}$, F1_{sts} e $\text{Revocação}_{\text{sb}}$. Os resultados serão analisados a partir de duas perspectivas: Resultados Gerais, e Top 5.

Os *datasets* utilizados neste trabalho são 10 estudos secundários, onde cada um inclui entre 15 e 152 estudos considerados relevantes. Os estudos considerados relevantes para um determinado *dataset* foi considerado o GS daquele estudo, se tornando o conjunto verdade do *dataset*. Os estudos são de temas variados, porém todos relacionados a computação, conforme demonstrado na Tabela 5.1. O critério utilizado para a seleção dos estudos secundários foi verificar se o GS do respectivo estudo secundário estava bem definido, visto que o processo experimental necessita do GS para o cálculo das métricas propostas. Os três estudos secundários utilizadas por Alves et al. (2022) estão inclusos na lista, portanto, foram adicionados sete estudos secundários para este trabalho. A decisão de incluir mais estudos secundários se deu com o

objetivo de avaliar a generalização dos resultados obtidos com a ferramenta para estudos secundários com diferentes tamanhos de GS ((quantidade de estudos relevantes incluídos), visto que os três utilizados por Alves et al. (2022) tinham tamanhos de GS variando somente de 15 a 46. Todos os sete estudos secundários adicionados possuem tamanho de GS maior que 46.

Para a realização dos experimentos foram coletados 10 estudos secundários, incluindo os três já utilizados por Alves et al. (2022) em seu experimento. Algumas características dos estudos utilizados podem ser vistas na Tabela 5.1.

Tabela 5.1: Estudos secundários utilizadas nos experimentos

Autor	Tamanho do GS	Tema
Abayomi-Alli et al. (2022)	56	Aprendizado Profundo
Azeem et al. (2019)	15	*Code smell*
Bertolino et al. (2019)	147	Testes de nuvem
Böhmer and Rinderle-Ma (2015)	152	Testes
van Dinter et al. (2021)	41	Automação de RSLs
Dissanayake et al. (2022)	72	Segurança
Cutigi Ferrari et al. (2018)	146	Testes de mutação
Hosseini et al. (2019)	46	Predição de defeitos
MOHAN et al. (2023)	76	Predição com IA
Vasconcellos et al. (2017)	30	Processos de Software

Inicialmente, o arquivo PDF de cada estudo primário incluído no estudo secundário foi coletado, juntamente com a extração dos metadados (título, resumo e palavras-chave) de cada estudo primário. Esse conjunto de estudos primários incluídos num estudo secundário foi chamado de GS. Após isso, os arquivos PDF foram transformados em arquivos de texto, de forma a possibilitar a criação do grafo de citação. Um grafo de citação indica quais estudos do GS um determinado estudo cita. Esse grafo de citação é usado posteriormente para auxiliar no cálculo das métricas de desempenho. Na figura 5.1 temos um exemplo de grafo de citação, onde uma aresta (A,B) indica que o estudo A cita o estudo B.

Conseguinte, foram determinados os conjuntos de parâmetros a serem utilizados para cada etapa da SeSGx. O LDA possui dois parâmetros recomendados por Alves et al. (2022): **frequência mínima de documentos**, e **número de tópicos**. Já na etapa de enriquecimento de termos, é possível ajustar somente o parâmetro **número de enriquecimentos**, que indica quantos "sinônimos" ou palavras similares devem ser concatenadas à um termo. Por fim, na etapa de formulação da *string*, é possível ajustar o parâmetro que indica o **número de palavras por tópico**.

Já os parâmetros do BERTopic são: **número de vizinhos**, utilizado no

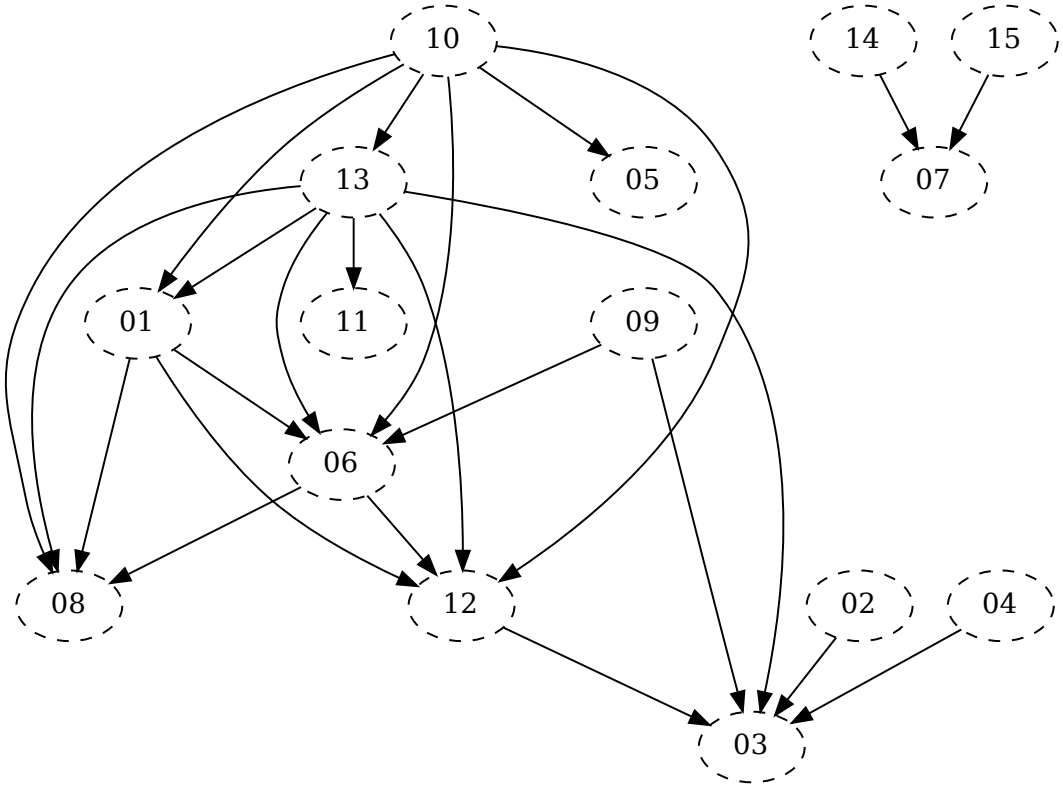

Figura 5.1: Exemplo de grafo de citação

algoritmo de redução de dimensionalidade, e **número de *clusters***, utilizado no K-means, que no fim das contas irá representar o número de tópicos.

As variações de parâmetros utilizadas foram as seguintes:

- **frequência mínima de documentos (LDA)**: [0.1, 0.2, 0.3, 0.4]
- **número de tópicos (LDA)**: [1, 2, 3, 4, 5]
- **número de vizinhos (BERTopic)**: [3, 5, 7]
- **número de *clusters* (BERTopic)**: [1, 2, 3, 4, 5]
- **número de enriquecimentos**: [0, 1, 2, 3]
- **número de palavras por tópico**: [5, 6, 7, 8, 9, 10]

Dessa forma, cada experimento foi executado utilizando o LDA e o BERTopic. Utilizamos força bruta para testar os parâmetros. Com isso, para obter a quantidade de combinações de parâmetros para cada abordagem, fizemos o produto cartesiano das listas de parâmetros de cada abordagem. As variações de parâmetros utilizadas foram as seguintes:

- LDA: 480 variações (produto das listas dos parâmetros: **frequência mínima de documentos**, **número de tópicos**, **número de enriquecimentos**, e **número de palavras por tópico**, portanto: $4 \times 5 \times 4 \times 6 = 480$)
- BERTopic: 360 variações (produto das listas dos paramêtros: **número de vizinhos**, **número de *clusters***, **número de enriquecimentos**, e **número de palavras por tópico**, portanto: $3 \times 5 \times 4 \times 6 = 360$)

Com isso, cada experimento gerou um total de $480 + 360 = 840$ *strings* de busca.

Com os paramêtros em mãos, e com o GS coletado, é iniciada a execução de um experimento. Um experimento inicia com a definição de um subconjunto dos estudos do GS, formado aleatoriamente. Esse subconjunto é chamado de *start set*. Para cada estudo secundário, foram executados 10 experimentos, ou seja, 10 *start sets* distintos para cada GS. Para este trabalho, a quantidade de estudos no *start set* equivale a 33% dos estudos do GS de cada estudo secundário. Esse valor foi definido após uma série de testes com diferentes quantidades de estudos no *start set*.

A execução de diversos experimentos com diferentes *start sets* é necessária para viabilizar a generalização dos resultados obtidos. Além disso, utilizar somente um *start set* pode trazer um viés muito grande, devido a possibilidade dele conter pelo menos um estudo de cada componente do grafo de citação, o que aumenta consideravelmente os valores de revocação considerando alguma técnica de *snowballing*.

O *start set* é então utilizado como entrada para extração de tópicos. Vale notar que nesta etapa, cada documento é representado somente pelo seu título, resumo, e palavras-chave. Após a extração de tópicos, é executado o enriquecimento de termos. Com os tópicos e as palavras enriquecidas, uma *string* de busca é formulada, sendo possível limitar o **número de palavras por tópico**. Um fator importante a ser notado é que devido ao fato de estarmos utilizando um estudo secundário publicado como fonte de um GS, houve a necessidade de incluirmos um operador de limite do ano de publicação, seja inferior ou superior, visando obter métricas condizentes com a realidade.

Com a *string* de busca formulada, é possível utilizá-la em uma máquina de busca. No caso desse trabalho, similarmente ao de Alves et al. (2022), foi utilizada a Scopus, devido à disponibilidade de uma API (*Application Programming Interface*) HTTP.

5.2 Exemplo de execução

Para exemplificar a execução da ferramenta, iremos propor um cenário onde um pesquisador quer desenvolver um estudo secundário e irá utilizar os

seguintes artigos como entrada para a ferramenta. É importante notar que todos os dados mencionados neste exemplo são hipotéticos.

- **Título**: Abordagens Automatizadas para a Detecção e Correção de Code Smells em Aplicações Web.

  Resumo: O estudo explora técnicas automatizadas de detecção e correção de code smells em aplicações web, comparando a eficácia de várias ferramentas disponíveis.

  Palavras-chave: code smells, detecção automatizada, correção de software, aplicações web.

- **Título**: Análise de Padrões de Code Smells em Projetos de Software Open Source

  Resumo: Este estudo investiga a prevalência e os tipos de code smells em repositórios de software open source, revelando correlações com falhas de software e manutenção.

  Palavras-chave: code smells, software open source, falhas de software, manutenção de software.

- **Título**: Impacto dos Code Smells na Manutenção e Evolução de Software: Um Estudo Longitudinal

  Resumo: Este artigo avalia como diferentes code smells influenciam a facilidade de manutenção e a evolução de software ao longo de um período de cinco anos.

  Palavras-chave: code smells, manutenção de software, evolução de software, estudo longitudinal.

A partir desses estudos, foram extraídos os seguintes tópicos (também hipotéticos):

- **Tópico 1**: code smells, detecção, correção
- **Tópico 2**: open source, falhas, manutenção
- **Tópico 3**: evolução, manutenção, code smells

Com os tópicos em mãos, a ferramenta realiza o enriquecimento dos termos:

- code smells → defeitos de código, anomalias de código
- detecção → identificação, descoberta
- correção → reparo, ajuste

- open source → código aberto, código livre
- falhas → defeitos, erros
- manutenção → conservação, suporte
- evolução → desenvolvimento, progresso

Com os tópicos e os termos enriquecidos, a ferramenta formula as *strings*:
Exemplo de uma *string* com 2 tópicos, 2 palavras por tópico, nenhum enriquecimento:

(code smells AND detecção)
OR
(open source AND falhas)

Exemplo de uma *string* com 2 tópicos, 2 palavras por tópico, 1 enriquecimento:

((code smells OR anomalias de código) AND (detecção OR descoberta))
OR
((open source OR código aberto) AND (falhas OR defeitos))

Por fim, a ferramenta utiliza as *strings* em um mecanismo de busca, como a Scopus, e utiliza os resultados retornados para calcular as métricas de completude.

5.3 Resultados

Visando medir o potencial máximo de cada estratégia, fizemos uma análise utilizando as cinco melhores *strings* em termos de $\text{Revocação}_{\text{sts}}$ de cada uma das implementações deste trabalho (SeSGx-BT e SeSGx-LDA). Além disso, também analisamos a média das métricas de $\text{Precisão}_{\text{sts}}$, $\text{Revocação}_{\text{sts}}$, F1_{sts}, e $\text{Revocação}_{\text{sb}}$ considerando todas as *strings* válidas de cada implementação, ou seja, somente *strings* que retornaram pelo menos um estudo quando aplicadas na Scopus.

Os dados de entrada para a ferramenta SeSGx são os QGS gerados aleatoriamente a partir do GS. De modo a evitar seleções enviesadas do QGS, as abordagens foram executadas 10 vezes, cada vez com um QGS diferente gerado aleatoriamente. Ao utilizar 10 QGS diferentes, com 10 estudos secundários diferentes, conseguimos obter resultados mais generalizáveis em comparação aos resultados obtidos por Alves et al. (2022), visto que os autores utilizaram somente três estudos secundários. Além disso, ao utilizar um

Tabela 5.2: Parte 1: Média das métricas obtidas com cada algoritmo

Estudo	Métrica	SeSGx-LDA	SeSGx-BT
Azeem	$\text{Precisão}_{\text{sts}}$	0.161 (± 0.272)	**0.279 (± 0.337)**
	$\text{Revocação}_{\text{sts}}$	**0.279 (± 0.201)**	0.243 (± 0.181)
	F1_{sts}	0.107 (± 0.143)	**0.134 (± 0.122)**
	$\text{Revocação}_{\text{sb}}$	0.776 (± 0.328)	**0.813 (± 0.298)**
Vasconcellos	$\text{Precisão}_{\text{sts}}$	0.074 (± 0.166)	**0.087 (± 0.187)**
	$\text{Revocação}_{\text{sts}}$	0.276 (± 0.211)	**0.277 (± 0.191)**
	F1_{sts}	**0.055 (± 0.067)**	0.054 (± 0.061)
	$\text{Revocação}_{\text{sb}}$	0.712 (± 0.315)	**0.778 (± 0.25)**
Dinter	$\text{Precisão}_{\text{sts}}$	0.021 (± 0.067)	**0.081 (± 0.175)**
	$\text{Revocação}_{\text{sts}}$	0.133 (± 0.154)	**0.16 (± 0.143)**
	F1_{sts}	0.022 (± 0.051)	**0.05 (± 0.076)**
	$\text{Revocação}_{\text{sb}}$	0.549 (± 0.405)	**0.663 (± 0.35)**
Hosseini	$\text{Precisão}_{\text{sts}}$	**0.121 (± 0.175)**	0.119 (± 0.241)
	$\text{Revocação}_{\text{sts}}$	**0.352 (± 0.16)**	0.204 (± 0.169)
	F1_{sts}	**0.114 (± 0.118)**	0.059 (± 0.052)
	$\text{Revocação}_{\text{sb}}$	**0.91 (± 0.159)**	0.893 (± 0.198)
Alli	$\text{Precisão}_{\text{sts}}$	0.038 (± 0.062)	**0.078 (± 0.104)**
	$\text{Revocação}_{\text{sts}}$	**0.278 (± 0.197)**	0.263 (± 0.186)
	F1_{sts}	0.046 (± 0.052)	**0.072 (± 0.063)**
	$\text{Revocação}_{\text{sb}}$	0.578 (± 0.229)	**0.582 (± 0.192)**

grupo experimental maior, conseguimos testar a performance das abordagens em diferentes ambientes, considerando as particularidades de cada estudo secundário, tal como tamanho do GS e domínio.

A discussão dos resultados será fortemente baseada na comparação entre as duas abordagens, destacando a melhoria de uma em relação a outra em termos de porcentagem.

5.3.1 Top 5 strings

Esta análise foi feita visando comparar o potencial máximo de cada abordagem nos diferentes datasets disponíveis. Para isso, analisamos a média das métricas das cinco melhores *strings* em termos de $\text{Revocação}_{\text{sts}}$.

Em termos de $\text{Precisão}_{\text{sts}}$, a SeSGx-LDA obteve melhores valores em 5 dos

Tabela 5.3: Parte 2: Média das métricas obtidas com cada algoritmo

Estudo	Métrica	SeSGx-LDA	SeSGx-BT
Dissanayake	$\text{Precisão}_{\text{sts}}$	0.02 (± 0.047)	**0.045 (± 0.102)**
	$\text{Revocação}_{\text{sts}}$	0.068 (± 0.085)	**0.076 (± 0.081)**
	F1_{sts}	0.018 (± 0.029)	**0.026 (± 0.035)**
	$\text{Revocação}_{\text{sb}}$	0.465 (± 0.339)	**0.514 (± 0.322)**
Mohan	$\text{Precisão}_{\text{sts}}$	0.012 (± 0.02)	**0.021 (± 0.034)**
	$\text{Revocação}_{\text{sts}}$	**0.176 (± 0.134)**	0.168 (± 0.12)
	F1_{sts}	0.02 (± 0.028)	**0.028 (± 0.03)**
	$\text{Revocação}_{\text{sb}}$	0.598 (± 0.254)	**0.642 (± 0.188)**
Ferrari	$\text{Precisão}_{\text{sts}}$	0.154 (± 0.127)	**0.212 (± 0.137)**
	$\text{Revocação}_{\text{sts}}$	0.297 (± 0.207)	**0.31 (± 0.215)**
	F1_{sts}	0.15 (± 0.104)	**0.18 (± 0.089)**
	$\text{Revocação}_{\text{sb}}$	0.975 (± 0.114)	**0.986 (± 0.048)**
Bertolino	$\text{Precisão}_{\text{sts}}$	0.052 (± 0.128)	**0.099 (± 0.191)**
	$\text{Revocação}_{\text{sts}}$	**0.08 (± 0.088)**	0.079 (± 0.077)
	F1_{sts}	0.027 (± 0.042)	**0.037 (± 0.052)**
	$\text{Revocação}_{\text{sb}}$	0.614 (± 0.304)	**0.652 (± 0.267)**
Bohmer	$\text{Precisão}_{\text{sts}}$	0.069 (± 0.129)	**0.08 (± 0.141)**
	$\text{Revocação}_{\text{sts}}$	**0.222 (± 0.137)**	0.196 (± 0.146)
	F1_{sts}	**0.068 (± 0.084)**	0.061 (± 0.071)
	$\text{Revocação}_{\text{sb}}$	**0.791 (± 0.213)**	0.788 (± 0.191)

Tabela 5.4: Parte 1: Média das métricas das 5 melhores strings em termos de Revocação$_{sts}$ obtidas com cada algoritmo

Estudo	Métrica	SeSGx-LDA	SeSGx-BT
Azeem	Precisão$_{sts}$	**0.023 (± 0.002)**	0.017 (± 0.004)
	Revocação$_{sts}$	**0.8 (± 0.0)**	0.733 (± 0.0)
	F1$_{sts}$	**0.044 (± 0.005)**	0.034 (± 0.007)
	Revocação$_{sb}$	1.0 (± 0.0)	1.0 (± 0.0)
Vasconcellos	Precisão$_{sts}$	0.005 (± 0.002)	**0.007 (± 0.003)**
	Revocação$_{sts}$	**0.753 (± 0.018)**	0.7 (± 0.0)
	F1$_{sts}$	0.011 (± 0.003)	**0.015 (± 0.005)**
	Revocação$_{sb}$	0.967 (± 0.0)	0.967 (± 0.0)
Dinter	Precisão$_{sts}$	**0.012 (± 0.004)**	0.006 (± 0.001)
	Revocação$_{sts}$	**0.727 (± 0.02)**	0.639 (± 0.027)
	F1$_{sts}$	**0.023 (± 0.007)**	0.012 (± 0.002)
	Revocação$_{sb}$	0.873 (± 0.044)	**0.898 (± 0.02)**
Hosseini	Precisão$_{sts}$	0.008 (± 0.002)	**0.01 (± 0.004)**
	Revocação$_{sts}$	**0.696 (± 0.0)**	0.635 (± 0.01)
	F1$_{sts}$	0.015 (± 0.004)	**0.02 (± 0.007)**
	Revocação$_{sb}$	0.939 (± 0.01)	**0.943 (± 0.012)**
Alli	Precisão$_{sts}$	0.013 (± 0.002)	**0.025 (± 0.016)**
	Revocação$_{sts}$	**0.836 (± 0.02)**	0.818 (± 0.015)
	F1$_{sts}$	0.025 (± 0.005)	**0.048 (± 0.029)**
	Revocação$_{sb}$	0.936 (± 0.03)	**0.943 (± 0.015)**

Tabela 5.5: Parte 2: Média das métricas das 5 melhores strings em termos de $\text{Revocação}_{\text{sts}}$ obtidas com cada algoritmo

Estudo	Métrica	SeSGx-LDA	SeSGx-BT
Dissanayake	$\text{Precisão}_{\text{sts}}$	**0.013 (± 0.004)**	0.007 (± 0.002)
	$\text{Revocação}_{\text{sts}}$	**0.547 (± 0.039)**	0.4 (± 0.028)
	F1_{sts}	**0.025 (± 0.008)**	0.014 (± 0.004)
	$\text{Revocação}_{\text{sb}}$	**0.903 (± 0.017)**	0.825 (± 0.043)
Mohan	$\text{Precisão}_{\text{sts}}$	**0.02 (± 0.01)**	0.019 (± 0.011)
	$\text{Revocação}_{\text{sts}}$	0.616 (± 0.025)	**0.629 (± 0.031)**
	F1_{sts}	**0.039 (± 0.019)**	0.037 (± 0.021)
	$\text{Revocação}_{\text{sb}}$	0.797 (± 0.015)	**0.821 (± 0.02)**
Ferrari	$\text{Precisão}_{\text{sts}}$	**0.059 (± 0.033)**	0.047 (± 0.006)
	$\text{Revocação}_{\text{sts}}$	0.859 (± 0.01)	0.859 (± 0.01)
	F1_{sts}	**0.109 (± 0.055)**	0.09 (± 0.01)
	$\text{Revocação}_{\text{sb}}$	1.0 (± 0.0)	1.0 (± 0.0)
Bertolino	$\text{Precisão}_{\text{sts}}$	0.056 (± 0.028)	**0.111 (± 0.009)**
	$\text{Revocação}_{\text{sts}}$	**0.552 (± 0.026)**	0.465 (± 0.031)
	F1_{sts}	0.099 (± 0.047)	**0.179 (± 0.013)**
	$\text{Revocação}_{\text{sb}}$	**0.91 (± 0.013)**	0.891 (± 0.008)
Bohmer	$\text{Precisão}_{\text{sts}}$	0.019 (± 0.002)	**0.027 (± 0.008)**
	$\text{Revocação}_{\text{sts}}$	0.566 (± 0.009)	**0.571 (± 0.023)**
	F1_{sts}	0.036 (± 0.004)	**0.051 (± 0.014)**
	$\text{Revocação}_{\text{sb}}$	**0.908 (± 0.012)**	0.861 (± 0.016)

10 datasets, com melhorias entre 4.25% e 86.86% em relação à SeSGx-BT. Nos 5 datasets restantes, a SeSGx-BT obteve melhorias entre 32.52% e 99.57%.

Ao analisar a Revocação$_{sts}$, vemos que a SeSGx-LDA obteve vantagem em 7 dos 10 datasets, com melhorias entre 2.18% e 36.81%. Em 2 dos 3 datasets restantes, a SeSGx-BT obteve aumentos entre 0.93% e 2.14%. No dataset Ferrari houve empate entre as abordagens.

Já em termos de F1$_{sts}$, a SeSGx-LDA obteve melhores valores em 5 dos 10 datasets, com melhorias entre 4.3% e 85.51%. Nos 5 datasets restantes, a SeSGx-BT obteve melhorias entre 31.71% e 91.9%.

Por fim, em termos de $Recall_{sb}$, a SeSGx-BT obteve melhores valores em 4 dos 10 datasets, com melhorias entre 0.46% e 2.97%. Em 3 dos 6 datasets restantes, a SeSGx-LDA obteve melhorias entre 2.14% e 9.43%. Nos 3 datasets restantes houve empate entre as abordagens.

Concluindo, é notável que a SeSGx-LDA, em sua performance máxima, obtém valores consideravelmente altos de Revocação$_{sts}$. No entanto, nas métricas de Precisão$_{sts}$ e $F1_{sts}$, apesar do empate no número de bases entre as abordagens, a SeSGx-BT mostrou melhorias mais expressivas (aumento médio de 52% considerando todas as bases), além de ter obtido vantagem sobre a SeSGx-LDA em uma base a mais em termos de Revocação$_{sb}$ (aumento médio de 4.9% considerando todas as bases).

5.3.2 Média e desvio padrão geral

Esta análise foi feita visando comparar o potencial médio e mais provável de cada abordagem nos diferentes datasets disponíveis. Para isso, utilizamos a média das métricas de todas as *strings* válidas de cada dataset, ou seja, das *strings* que retornaram pelo menos um resultado na busca em base.

Em termos de Precisão$_{sts}$, a SeSGx-BT obteve as melhores médias em 9 dos 10 datasets, obtendo valores entre 16.52% a 286.25% melhores que a SeSGx-LDA. No dataset Hosseini a SeSGx-LDA foi 1.59% melhor que a SeSGx-BT.

Por outro lado, na Revocação$_{sts}$, a SeSGx-LDA obteve melhores valores em 6 dos 10 datasets, com melhorias entre 0.32% e 20% em relação a SeSGx-BT. Nos 4 datasets restantes, a SeSGx-BT obteve valores entre 0.32% e 20% melhores que a SeSGx-LDA.

Com isso, a SeSGx-BT obteve uma melhor performance em 7 dos 10 datasets em termos de F1$_{sts}$, com melhorias entre 20.5% e 126.38%. Nos 3 datasets restantes a SeSGx-LDA obteve melhorias entre 10.87% e 94.07%.

Por fim, em termos de Revocação$_{sb}$, a SeSGx-BT obteve vantagem em 8 dos 10 datasets, com melhorias entre 0.8% e 20.76%. Nos 2 datasets restantes, a SeSGx-LDA obteve melhorias entre 0.32% e 1.92%.

Concluindo, os grandes aumento na Precisão$_{sts}$ e Revocação$_{sb}$ obtidos pela

SeSGx-BT mostram, respectivamente, que o esforço do pesquisador pode ser reduzido na etapa de seleção de estudos devido ao menor número de estudos encontrados na busca automatizada, e que a SeSGx-BT é capaz de formular *strings* mais performáticas para estudos secundários com busca automatizada seguida de *snowballing*. Além disso, o ganho de performance da SeSGx-LDA em termos de $\text{Revocação}_{\text{sts}}$ não foi tão expressivo quanto o ganho de $\text{Precisão}_{\text{sts}}$ obtido pela SeSGx-BT, fazendo com que o F1_{sts} da SeSGx-LDA fosse melhor. No entanto, é importante notar que a SeSGx-LDA pode ser uma melhor alternativa para estudos secundários baseados somente em busca automatizada, e focados em completude, ao custo de uma precisão reduzida.

5.4 Ablação

Visando avaliar o impacto dos parâmetros de geração das *strings* de busca, que são utilizados tanto na SeSGx-LDA quanto na SeSGx-BT, executamos um procedimento de ablação para os parâmetros de número de enriquecimentos e número de palavras por tópico.

Para isso, fixamos os parâmetros específicos de cada abordagem nos valores de suas respectivas modas estatísticas, para cada estratégia e cada estudo secundário. A moda foi calculada utilizando as 35 melhores *strings* de cada experimento em termos de F1_{sts}, totalizando 350 *strings* de cada estudo secundário. Decidimos utilizar 35 *strings* por experimento pois essa quantidade representa aproximadamente 10% do total de strings geradas por experimento. Então, utilizamos a moda para filtrar os resultados gerais. É importante notar que ao analisar um determinado parâmetro (por exemplo, o número de enriquecimentos), o outro parâmetro (nesse caso, o número de palavras por tópico) também será fixado utilizando sua respectiva moda.

Com os dados filtrados conseguimos analisar o impacto do valor de cada parâmetro nas métricas de $\text{Precisão}_{\text{sts}}$, $\text{Revocação}_{\text{sts}}$ e $\text{Revocação}_{\text{sb}}$ em cada estudo secundário. As Figuras 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, 5.11 mostram os gráficos para o número de enriquecimentos, e as Figuras 5.12, 5.13, 5.14, 5.15, 5.16, 5.17, 5.18, 5.19, 5.20, 5.21 mostram os gráficos para o número de palavras por tópico.

Os gráficos do número de enriquecimentos nos mostram que, para a maioria dos estudos secundários, ao aumentar o número de enriquecimentos, a $\text{Precisão}_{\text{sts}}$ diminui. Isso é esperado devido a estratégia que utilizamos para formular a *string*, que consiste em concatenar os termos enriquecidos utilizando um operador de adição, nesse caso o operador “OR”. O uso desse operador expande a busca, retornando mais resultados, portanto reduzindo a precisão. Também notamos que a $\text{Revocação}_{\text{sts}}$ e a $\text{Revocação}_{\text{sb}}$ tendem

a aumentar de acordo com o número de enriquecimentos. Esse comportamento também é esperado devido à expansão da busca com o operador "OR", retornando mais estudos relevantes, portanto aumentando a revocação. É importante notar que mesmo nos estudos onde há uma queda nos valores de revocação, os valores de revocação para *strings* com pelo menos um enriquecimento costumam ser maiores que *strings* com nenhum enriquecimento.

Por outro lado, os gráficos do número de palavras por tópico nos mostram que a $\text{Precisão}_{\text{sts}}$ aumenta de acordo com o número de palavras por tópico. Esse comportamento se dá pela forma como concatenamos as palavras de um mesmo tópico usando um operador de intersecção, nesse caso o operador "AND". Isso irá restringir a busca, portanto retornando menos resultados no geral, o que aumenta a precisão. Além disso, a revocação no geral diminui conforme o número de palavras por tópico aumenta. Isso ocorre devido a restrição que ocorre na busca, que ao retornar menos estudos no geral, também acaba por não retornar alguns estudos relevantes.

Com isso em mente, podemos concluir que:

- Aumentar o número de enriquecimentos implica em encontrar mais estudos relevantes, ao custo de reduzir a precisão, o que aumenta o esforço do pesquisador durante a seleção de estudos.
- Aumentar o número de palavras por tópico implica em *strings* com maior precisão, reduzindo o esforço durante a seleção de estudos, ao custo de uma revocação reduzida, impactando negativamente a completude da busca.

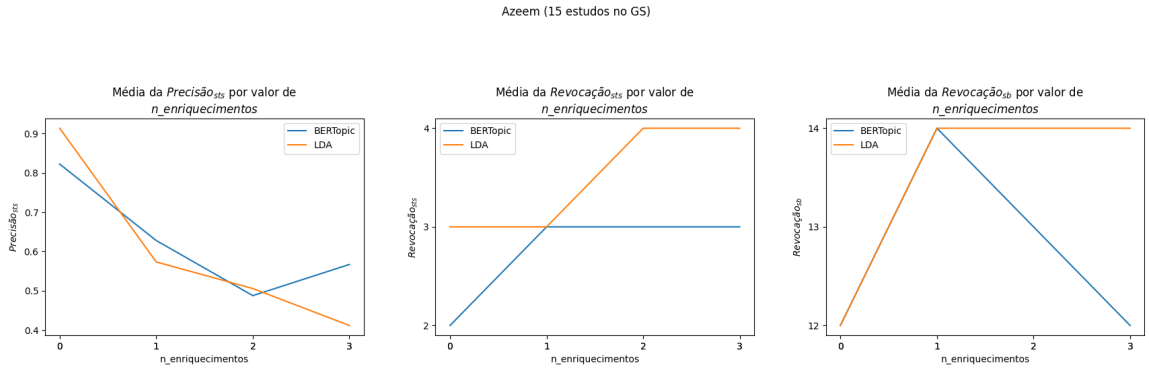

Figura 5.2: Média das métricas por número de enriquecimentos no dataset Azeem

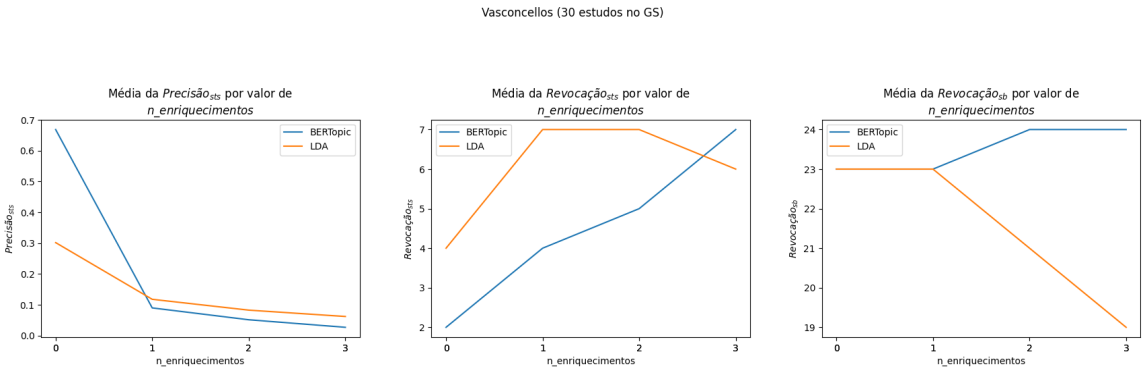

Figura 5.3: Média das métricas por número de enriquecimentos no dataset Vasconcellos

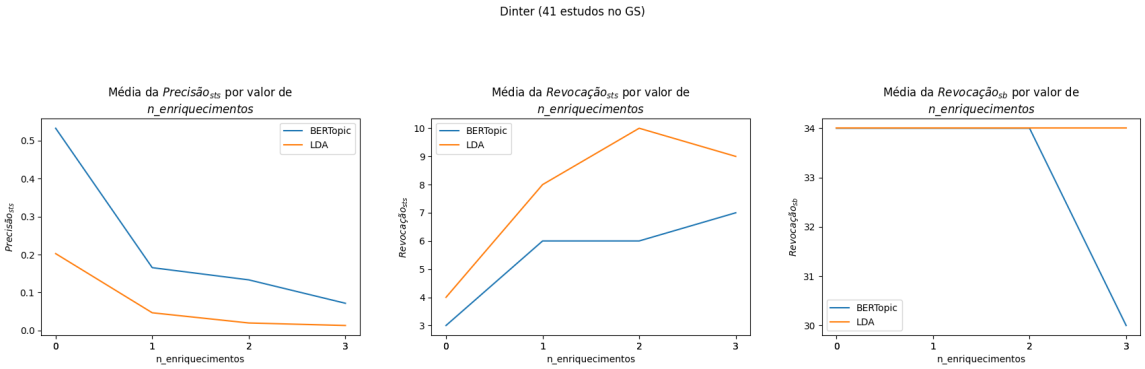

Figura 5.4: Média das métricas por número de enriquecimentos no dataset Dinter

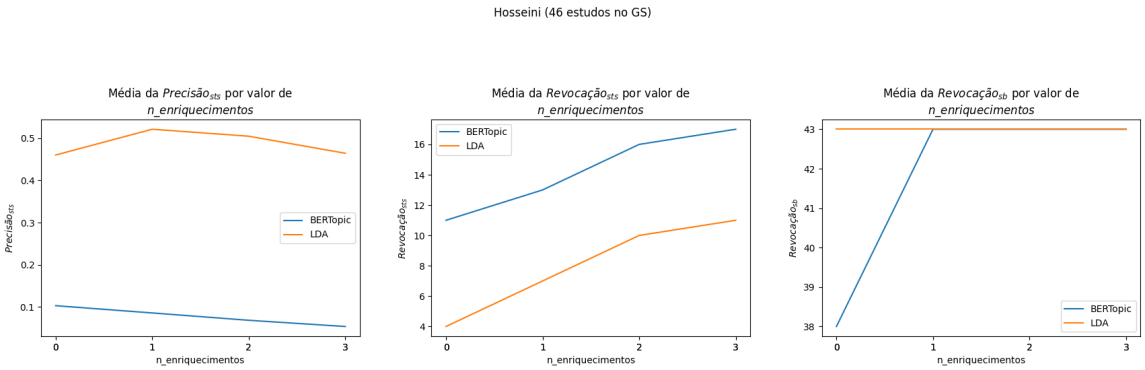

Figura 5.5: Média das métricas por número de enriquecimentos no dataset Hosseini

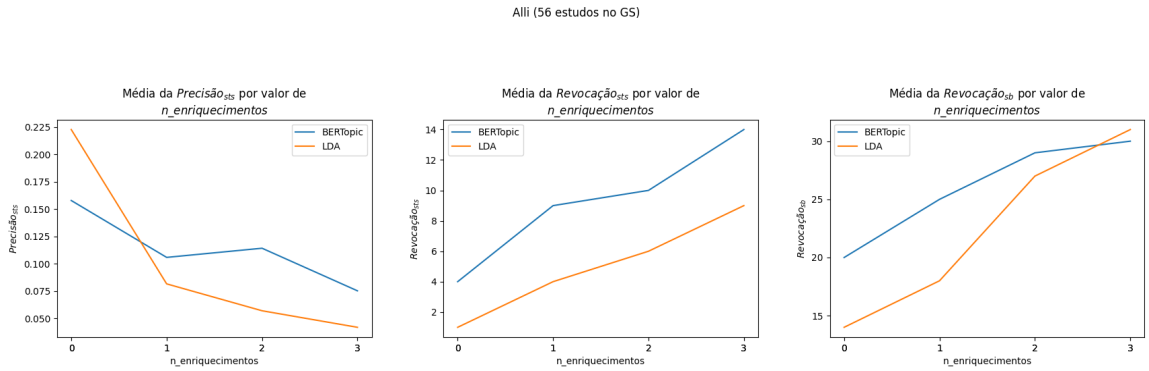

Figura 5.6: Média das métricas por número de enriquecimentos no dataset Alli

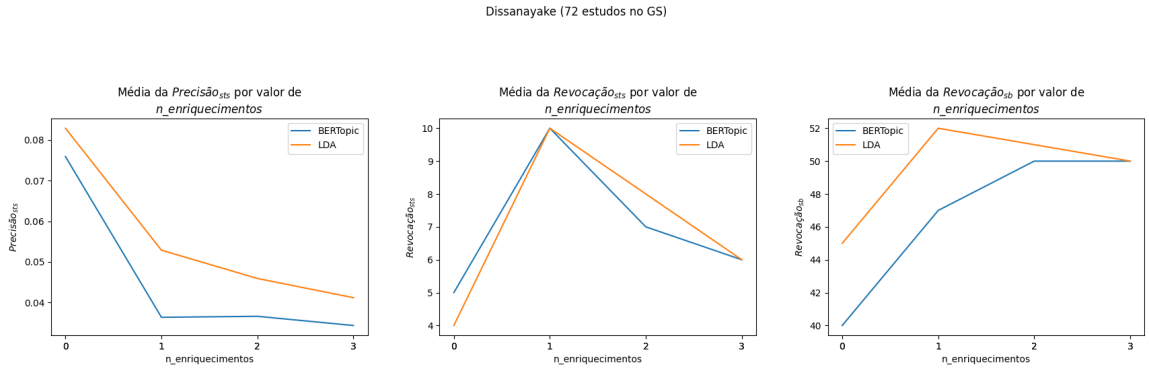

Figura 5.7: Média das métricas por número de enriquecimentos no dataset Dissanayake

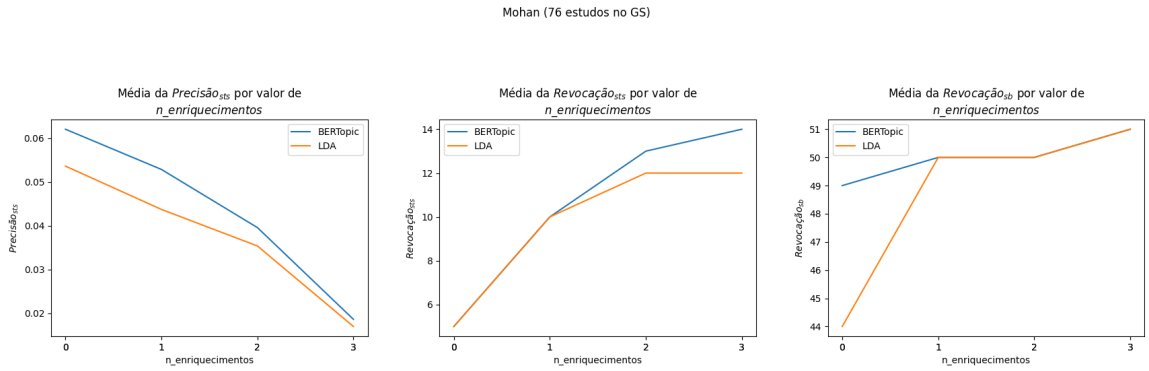

Figura 5.8: Média das métricas por número de enriquecimentos no dataset Mohan

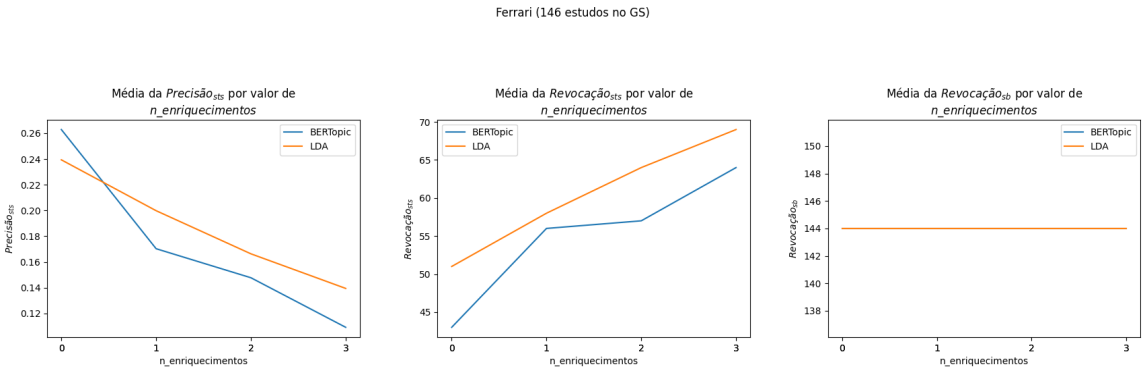

Figura 5.9: Média das métricas por número de enriquecimentos no dataset Ferrari

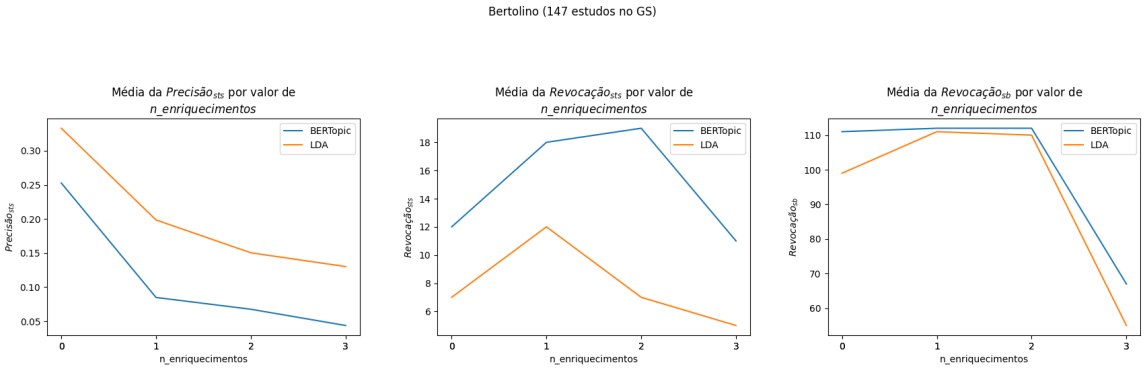

Figura 5.10: Média das métricas por número de enriquecimentos no dataset Bertolino

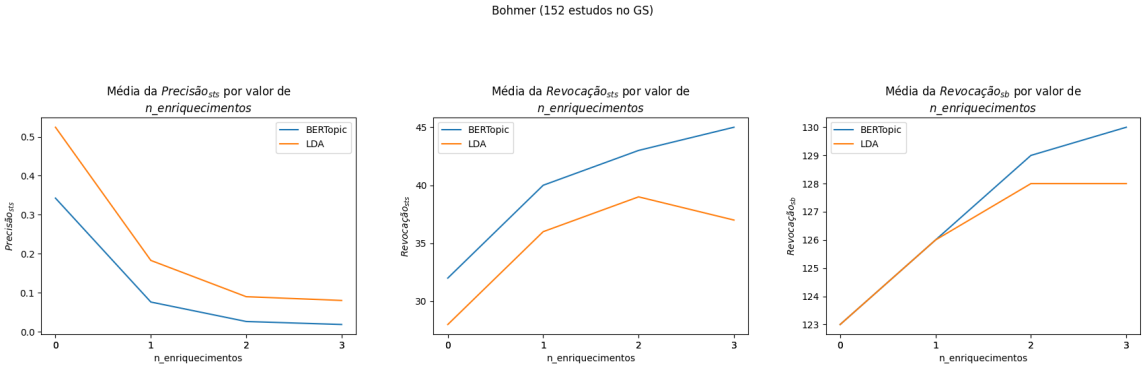

Figura 5.11: Média das métricas por número de enriquecimentos no dataset Bohmer

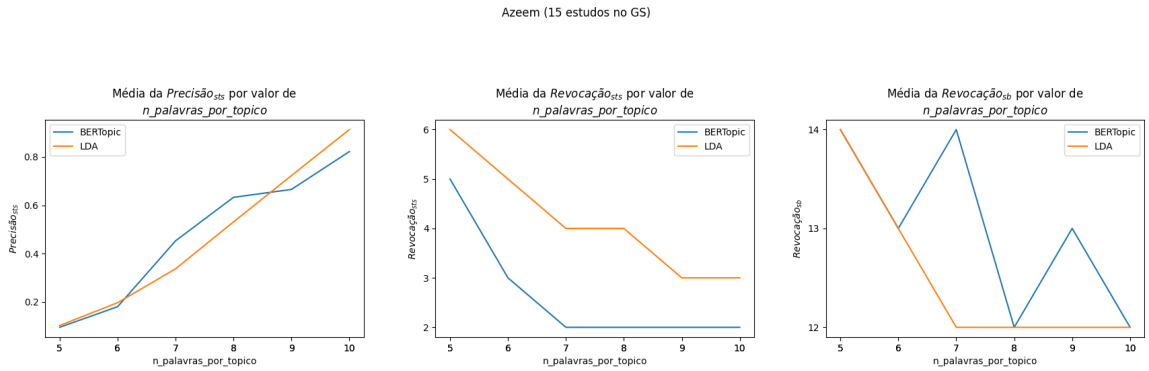

Figura 5.12: Média das métricas por número de palavras por tópico no dataset Azeem

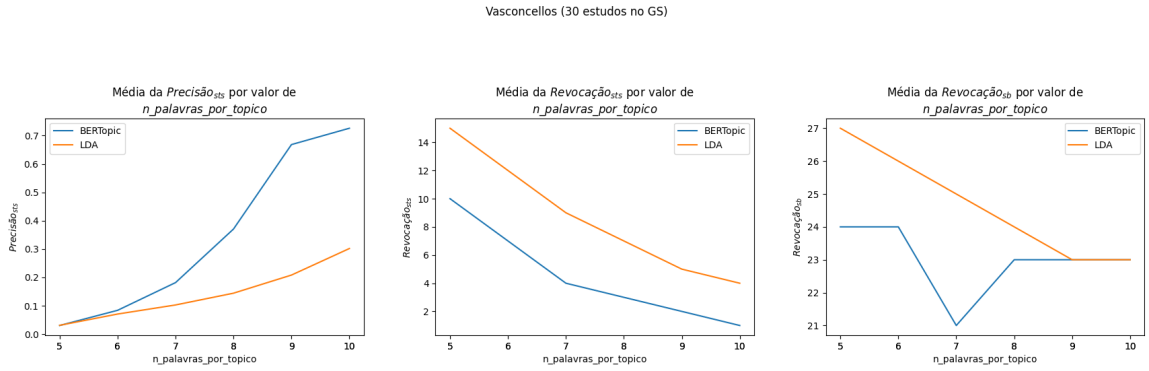

Figura 5.13: Média das métricas por número de palavras por tópico no dataset Vasconcellos

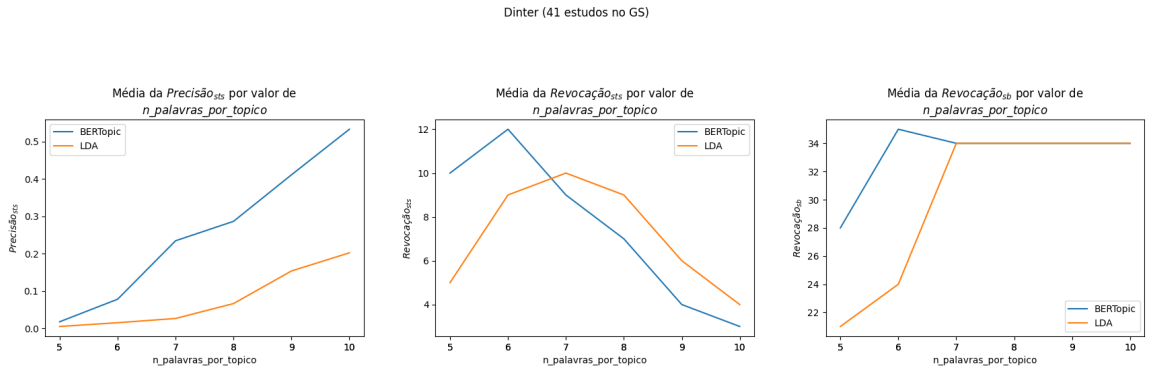

Figura 5.14: Média das métricas por número de palavras por tópico no dataset Dinter

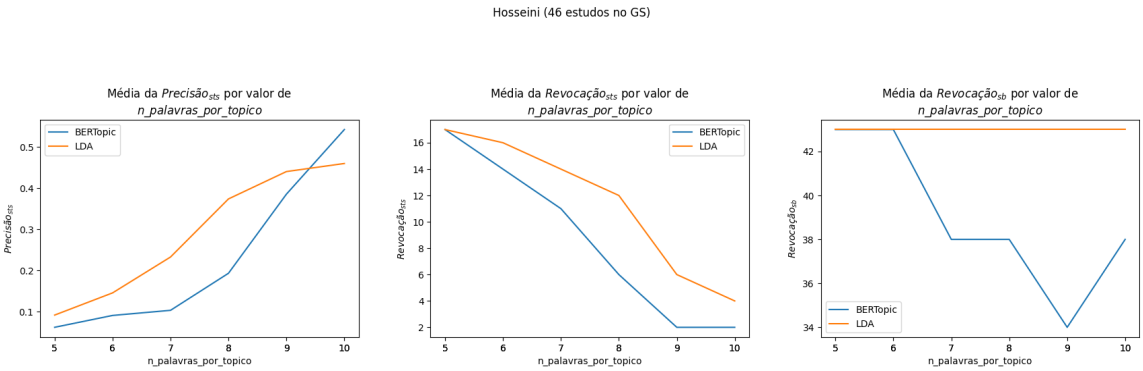

Figura 5.15: Média das métricas por número de palavras por tópico no dataset Hosseini

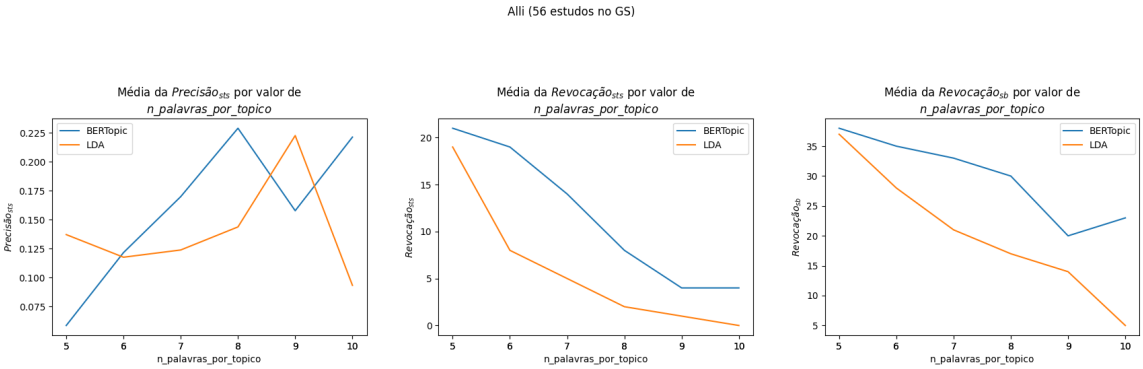

Figura 5.16: Média das métricas por número de palavras por tópico no dataset Alli

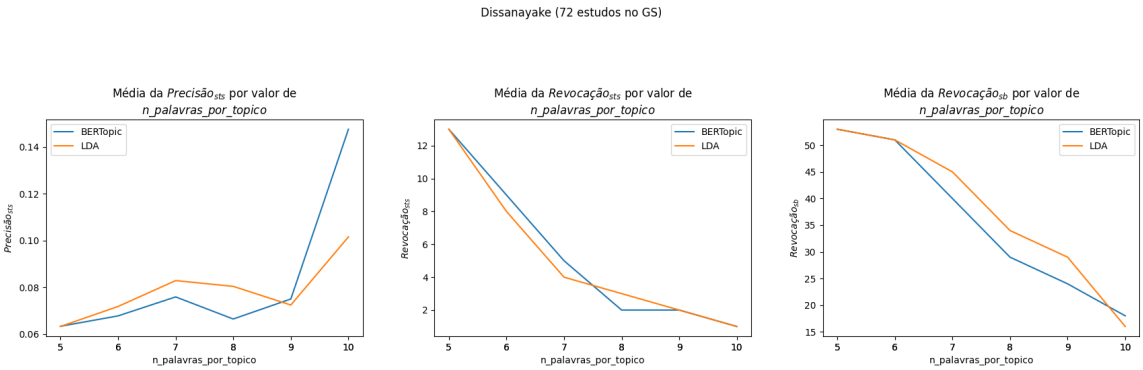

Figura 5.17: Média das métricas por número de palavras por tópico no dataset Dissanayake

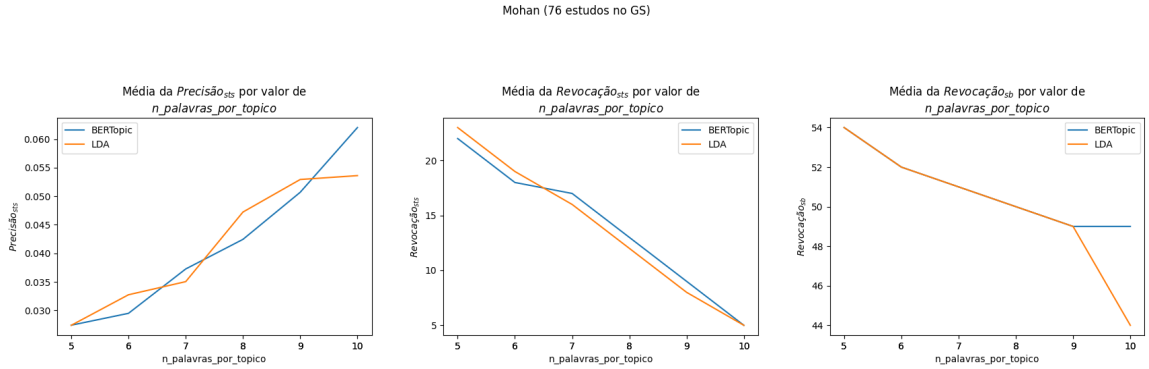

Figura 5.18: Média das métricas por número de palavras por tópico no dataset Mohan

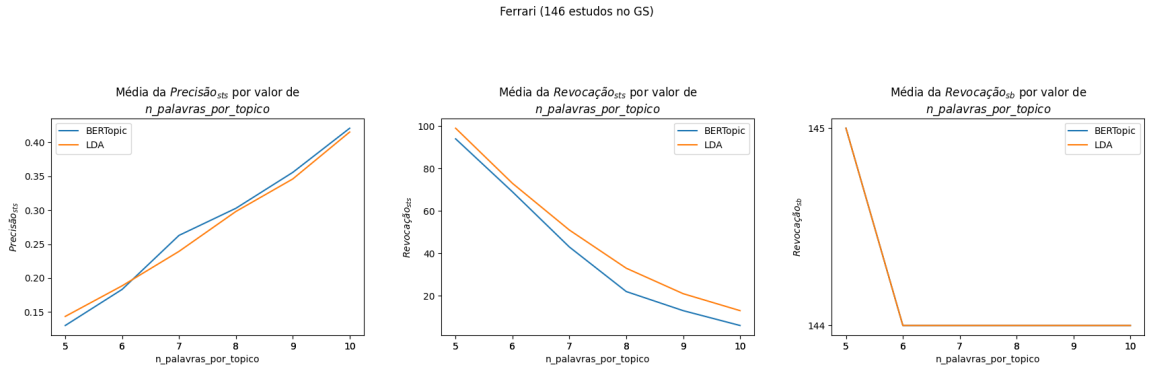

Figura 5.19: Média das métricas por número de palavras por tópico no dataset Ferrari

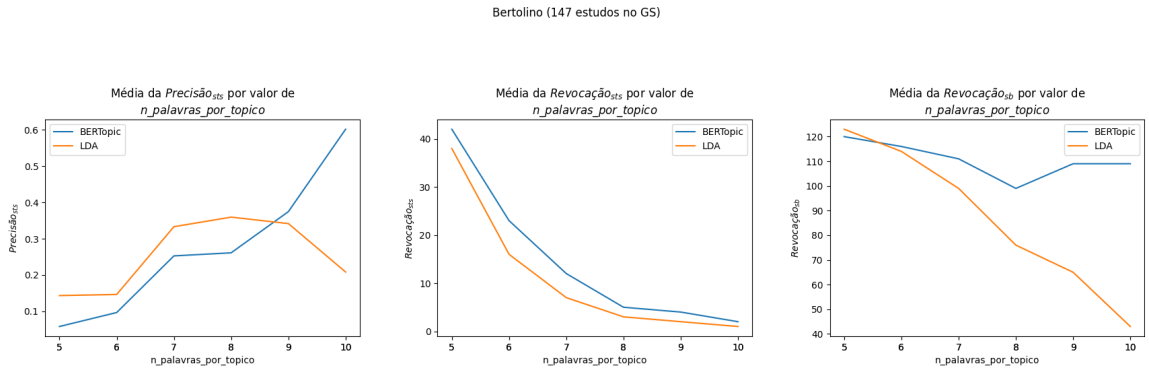

Figura 5.20: Média das métricas por número de palavras por tópico no dataset Bertolino

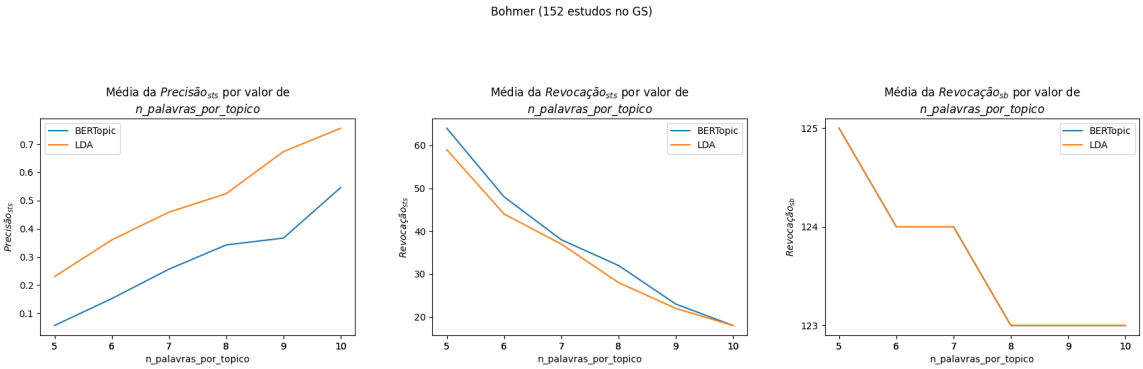

Figura 5.21: Média das métricas por número de palavras por tópico no dataset Bohmer

CAPÍTULO

6

Conclusão

Neste trabalho propomos uma ferramenta que facilita a experimentação de novos algoritmos, baseada na abordagem proposta por Alves et al. (2022), chamada SeSGx. Também fornecemos evidências acerca da performance de modelos de linguagem baseados em *deep learning* no contexto de estudos secundários, especificamente quando utilizados em algoritmos de extração de tópicos, como o BERTopic.

Nossa abordagem baseada em *deep learning*, SeSGx-BT, superou a abordagem estatística SeSGx-LDA em termos de média geral nos seguintes cenários: 9 de 10 datasets em termos de $\text{Precisão}_{\text{sts}}$, com aumentos entre 16% e 268%; 7 de 10 datasets em termos de F1_{sts}, com aumentos entre 20% e 126%; e 8 de 10 datasets em termos de $\text{Revocação}_{\text{sb}}$, com aumentos entre 0.8% e 20%. Em termos de $\text{Revocação}_{\text{sts}}$, a SeSGx-LDA superou a SeSGx-BT em 6 de 10 datasets, com aumentos entre 1% e 72%.

Esses resultados indicam que as *embeddings* contextuais presentes em modelos de linguagem baseados em *deep learning* podem melhorar a performance de algoritmos de extração de tópicos, como visto com o BERTopic. O aumento na $\text{Precisão}_{\text{sts}}$ mostra que os tópicos encontrados pelo BERTopic são capazes de descrever os documentos com uma maior precisão, portanto retornando menos estudos irrelevantes. Com isso, notamos que o $\text{Revocação}_{\text{sts}}$ não foi ótimo, devido ao menor número de estudos relevantes retornados pela busca. No entando, o F1_{sts} nos mostra que o ganho de precisão foi mais significante que a perda de revocação. Por fim, o aumento na $\text{Revocação}_{\text{sb}}$ pode indicar que, por mais que a SeSGx-BT não tenha encontrado tantos estudos relevantes durante a busca automatizada (baixa $\text{Revocação}_{\text{sts}}$), os estudos relevantes que foram encontrados possuem mais conexões com outros estudos

relevantes por meio de citações.

Respondendo às questões de pesquisa, temos o seguinte:

1. O BERTopic é capaz de gerar tópicos que consigam desempenho melhor que o LDA em termos de precisão e revocação na geração automática de *strings* de busca?

Considerando a média geral dos resultados, o BERTopic superou o LDA de maneira expressiva em termos de $\text{Precisão}_{\text{sts}}$. Já em termos de revocação, o BERTopic superou o LDA apenas na $\text{Revocação}_{\text{sts}}$.

2. Métodos baseados em aprendizado profundo para a geração de tópicos, como o BERTopic, geram resultados com maior consistência na geração de *strings* de busca - isto é, menor variância em diferentes bases de dados?

Considerando a média geral, o BERTopic foi consistente em obter melhores médias de $\text{Precisão}_{\text{sts}}$, F1_{sts}, e $\text{Revocação}_{\text{sb}}$ em grande parte dos datasets. No entanto, o desvio padrão do BERTopic em termos de $\text{Precisão}_{\text{sts}}$ mostra uma maior variação dos resultados. Nas outras medidas o desvio padrão do BERTopic e do LDA foram similares.

3. Na geração de *strings* de busca, quais parâmetros possuem maior influência nos resultados obtidos?

A partir dos dados obtidos por meio da ablação, podemos inferir que tanto o **número de enriquecimentos** quanto o **número de palavras por tópico**, que são os parâmetros compartilhados entre as abordagens, influenciam os resultados de maneiras diferentes. De um modo geral, aumentar o número de enriquecimentos aumenta o número de estudos relevantes encontrados (maior revocação), ao custo de uma menor precisão devido ao maior ruído. Por outro lado, aumentar o número de palavras por tópico aumenta a precisão devido à maior especificidade da *string*, ao custo de uma menor revocação, visto que menos estudos serão retornados pela busca.

6.1 Limitações

Uma observação importante é que, apesar de um experimento possuir 840 variações de parâmetros (480 para o LDA + 360 para o BERTopic), é possível que algumas da *strings* sejam exatamente iguais, mesmo que tenham sido geradas a partir de parâmetros diferentes. Após uma análise dos resultados, o que pudemos concluir foi que, quanto menor o *start set*, maior a

chance de geração de *strings* duplicadas. Estimamos que cerca de 18% do total de *strings* geradas eram duplicatas. No total, foram geradas 84000 *strings* (840 por experimento × 10 experimentos por RSL × 10 RSLs), incluindo duplicatas.

Além disso, algumas das *strings* de busca geradas não tiveram seu desempenho avaliado, por algum dos seguintes motivos:

- **Não processável pela Scopus**: Algumas *strings* geradas pela SeSGx continham muitos caracteres, de modo que a Scopus não fosse capaz de processar a *string* de busca, resultando em uma exceção.
- **Nenhum resultado encontrado**: Algumas *strings* não retornaram nenhum resultado.

No entanto, estimamos que somente 3% do total de *strings* únicas sofreram algum desses impasses. Cerca de 2.3% eram não processáveis, e os 0.7% restantes não retornaram nenhum resultado. Essas *strings* foram desconsideradas para os cálculos de média e desvio padrão.

O grafo de citação é gerado utilizando técnicas de distância de edição (Levenshtein et al., 1966) e correspondência difusa (*fuzzy matching*). Para isso, o arquivo PDF de um determinado estudo A é transformado em um arquivo de texto, e então, cada um dos títulos dos estudos do GS é buscado nesse arquivo de texto, de forma a encontrar quais são as referências de A (ou quais estudos são citados por A). Dessa forma, é montado o grafo de citação baseado em *backward snowballing* (BSB). Para a montagem do grafo de citação baseado em *snowballing* completo (BSB + FSB), o grafo de citação baseado em BSB tem suas arestas direcionadas transformadas em não direcionadas, ou seja, assumimos que, se A cita B, então B é citado por A, e que é possível encontrar A ou B partindo de qualquer um dos dois estudos.

Todavia, pode não ser possível encontrar A a partir de B (ou seja, via FSB em B), visto que é necessário que os indexadores possuam esse registro. Isso é uma limitação conhecida do método de experimentação utilizado. Não foi possível a montagem de um grafo de citação baseado somente em FSB devido a ausência da possibilidade de automação dessa técnica. Não foram encontrados métodos eficientes e factíveis para realizar a coleta desses dados.

A quantidade limitada de documentos do dataset Azeem não permitia que o BERTopic atingisse um funcionamento correto, resultando em erros de execução. Para resolver esse problema, o QGS gerado a partir do dataset Azeem foi duplicado. Ademais, para evitar vantagens ou desvantagens para o BERTopic devido à essa duplicação, o QGS também foi duplicado para o LDA.

No dataset Bohmer, os autores documentam a existência de 153 estudos em seu GS. No entanto, durante a preparação dos dados para uso na SeSGx,

notamos que os autores haviam duplicado um dos estudos em sua contagem. Manter esse valor incorreto causaria problemas no processo de cálculo das métricas pela SeSGx. Para corrigir esse problema, removemos um dos estudos duplicados, portanto, consideramos que o dataset Bohmer possui 152 estudos em seu GS.

6.2 Dificuldades

A implementação da abordagem disponibilizada por Alves et al. (2022) possui um tempo de execução muito longo. Devido ao maior número de bases de dados utilizadas neste trabalho, foi necessária uma refatoração completa da ferramenta.

Entender e generalizar a abordagem para que ela funcionasse com outros algoritmos nas etapas de extração de tópicos e de enriquecimento foi uma tarefa difícil. A refatoração incluiu: programação paralela para tarefas intensivas na CPU (*CPU Bound*, como a geração do grafo de citação; programação concorrente para tarefas intensivas em entradas e saídas (*I/O Bound*), como chamadas HTTP para a API da base indexadora utilizada; sistemas de cache para evitar chamadas aos modelos de extração de tópicos e de enriquecimento de termos; armazenamento dos resultados em um banco de dados relacional, facilitando a consulta; e uma nova arquitetura que facilita a manutenção.

Após a refatoração, reduzimos o tempo de execução da ferramenta de aproximadamente 24 horas para 2 horas.

6.3 Trabalhos futuros

O BERTopic possui uma estrutura modular. Isso significa que diferentes estratégias podem ser utilizadas em cada etapa. Neste trabalho, substituímos o algoritmo de agrupamento padrão do BERTopic (HDBSCAN) pelo K-means. Outras estratégias serão testadas em diferentes etapas do BERTopic, como por exemplo, alterar o modelo utilizado para geração de *embeddings*.

A ferramenta SeSGx foi criada visando a experimentação de novos algoritmos utilizando a abordagem proposta por Alves et al. (2022). No entanto, a ferramenta ainda encontra-se em estado experimental, o que a torna pouco viável para usuários finais, sendo estes os pesquisadores que desenvolvem estudos secundários. Como trabalho futuro, a SeSGx será reimplementada visando o usuário final.

A SeSGx possui a etapa de enriquecimento de termos, onde utilizamos apenas o BERT. Essa etapa, além da etapa de extração de tópicos, impacta diretamente a performance das *strings*, nos mostrando que novas estratégias

devem ser testadas nessa etapa, visando melhora os resultados.

6.4 Disponibilidade dos artefatos

Criamos uma organização[1] no GitHub onde a ferramenta[2] está disponível, assim como um programa[3] executado via interface de linha de comando, que foi utilizado para a execução dos experimentos.

[1] https://github.com/sesgx
[2] https://github.com/sesgx/sesgx
[3] https://github.com/sesgx/sesgx-cli

Referências Bibliográficas

Abayomi-Alli, O. O., Damaševičius, R., Qazi, A., Adedoyin-Olowe, M., e Misra, S. (2022). Data augmentation and deep learning methods in sound classification: A systematic review. *Electronics*, 11(22).

Adeva, J. G., Atxa, J. P., Carrillo, M. U., e Zengotitabengoa, E. A. (2014). Automatic text classification to support systematic reviews in medicine. *Expert Systems with Applications*, 41(4):1498–1508.

Aggarwal, C. C., Hinneburg, A., e Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. In *Database Theory—ICDT 2001: 8th International Conference London, UK, January 4–6, 2001 Proceedings 8*, páginas 420–434. Springer.

Alves, L. F., Vasconcellos, F. J., e Nogueira, B. M. (2022). Sesg: a search string generator for secondary studies with hybrid search strategies using text mining. *Empirical Software Engineering*, 27(5):105.

Azeem, M. I., Palomba, F., Shi, L., e Wang, Q. (2019). Machine learning techniques for code smell detection: A systematic literature review and meta-analysis. *Information and Software Technology*, 108:115–138.

Badampudi, D., Wohlin, C., e Petersen, K. (2015). Experiences from using snowballing and database searches in systematic literature studies. In *Proceedings of the 19th international conference on evaluation and assessment in software engineering*, páginas 1–10.

Bannach-Brown, A., Przybyła, P., Thomas, J., Rice, A. S., Ananiadou, S., Liao, J., e Macleod, M. R. (2019). Machine learning algorithms for systematic review: reducing workload in a preclinical review of animal studies and reducing human screening error. *Systematic reviews*, 8(1):1–12.

Bengio, Y., Ducharme, R., e Vincent, P. (2000). A neural probabilistic language model. *Advances in neural information processing systems*, 13.

Bertolino, A., Angelis, G. D., Gallego, M., García, B., Gortázar, F., Lonetti, F., e Marchetti, E. (2019). A systematic review on cloud testing. *ACM Comput. Surv.*, 52(5).

Beyer, K., Goldstein, J., Ramakrishnan, R., e Shaft, U. (1999). When is "nearest neighbor" meaningful? In *Database Theory—ICDT'99: 7th International Conference Jerusalem, Israel, January 10–12, 1999 Proceedings 7*, páginas 217–235. Springer.

Blei, D. M., Ng, A. Y., e Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Bojanowski, P., Grave, E., Joulin, A., e Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Böhmer, K. e Rinderle-Ma, S. (2015). A systematic literature review on process model testing: Approaches, challenges, and research directions.

Collobert, R. e Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, páginas 160–167.

Cutigi Ferrari, F., Viola Pizzoleto, A., e Offutt, J. (2018). A systematic review of cost reduction techniques for mutation testing: Preliminary results. In *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, páginas 1–10.

Devlin, J., Chang, M.-W., Lee, K., e Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Dickersin, K., Scherer, R., e Lefebvre, C. (1994). Systematic reviews: identifying relevant studies for systematic reviews. *Bmj*, 309(6964):1286–1291.

Dieste, O., Grimán, A., e Juristo, N. (2009). Developing search strategies for detecting relevant experiments. *Empirical Software Engineering*, 14:513–539.

Dissanayake, N., Jayatilaka, A., Zahedi, M., e Babar, M. A. (2022). Software security patch management - a systematic literature review of challenges, approaches, tools and practices. *Information and Software Technology*, 144:106771.

Dybå, T. e Dingsøyr, T. (2008). Strength of evidence in systematic reviews in software engineering. In *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, páginas 178–187.

Ebecken, N. F., Lopes, M. C. S., COSTA, M. C., et al. (2003). Mineração de textos. *Sistemas inteligentes: fundamentos e aplicações. São Carlos: Manole*, páginas 337–370.

Grootendorst, M. (2022). Bertopic: Neural topic modeling with a class-based tf-idf procedure.

Hausner, E., Waffenschmidt, S., Kaiser, T., e Simon, M. (2012). Routine development of objectively derived search strategies. *Systematic reviews*, 1:1–10.

Hosseini, S., Turhan, B., e Gunarathna, D. (2019). A systematic literature review and meta-analysis on cross project defect prediction. *IEEE Transactions on Software Engineering*, 45(2):111–147.

Imtiaz, S., Bano, M., Ikram, N., e Niazi, M. (2013). A tertiary study: experiences of conducting systematic literature reviews in software engineering. In *Proceedings of the 17th international conference on evaluation and assessment in software engineering*, páginas 177–182.

Indurkhya, N. e Damerau, F. J. (2010). *Handbook of Natural Language Processing*. Chapman & Hall/CRC, 2nd edition.

Joulin, A., Grave, E., Bojanowski, P., e Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(2004):1–26.

Kitchenham, B. A., Budgen, D., e Brereton, P. (2015). *Evidence-based software engineering and systematic reviews*, volume 4. CRC press.

Kontonatsios, G., Spencer, S., Matthew, P., e Korkontzelos, I. (2020). Using a neural network-based feature extraction method to facilitate citation screening for systematic reviews. *Expert Systems with Applications: X*, 6:100030.

Kuhrmann, M., Fernández, D. M., e Daneva, M. (2017). On the pragmatic design of literature studies in software engineering: an experience-based guideline. *Empirical software engineering*, 22:2852–2891.

Levenshtein, V. I. et al. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, páginas 707–710. Soviet Union.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In Cam, L. M. L. e Neyman, J., editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, páginas 281–297. University of California Press.

Mergel, G. D., Silveira, M. S., e da Silva, T. S. (2015). A method to support search string building in systematic literature reviews through visual text mining. In *Proceedings of the 30th annual ACM symposium on applied computing*, páginas 1594–1601.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., e Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

MOHAN, V., Ali, A.-F. M., e Ameedeen, M. A. (2023). A systematic survey on the research of ai-predictive models for wastewater treatment processes. *Iraqi Journal For Computer Science and Mathematics*, 4(1):102–113.

Moura, M. (2006). Uma abordagem para a construçao e atualizaçao de taxonomias de tópicos a partir de coleções de textos dinâmicas. *Inistituto de Ciências Matemáticas e de Computaçao - ICMC/USP*.

Mourão, E., Pimentel, J. F., Murta, L., Kalinowski, M., Mendes, E., e Wohlin, C. (2020). On the performance of hybrid search strategies for systematic literature reviews in software engineering. *Information and software technology*, 123:106294.

Rezende, S. O., Pugliesi, J. B., Melanda, E. A., e Paula, M. d. (2003). Mineração de dados. *Sistemas inteligentes: fundamentos e aplicações*, 1:307–335.

Riaz, M., Sulayman, M., Salleh, N., e Mendes, E. (2010). Experiences conducting systematic reviews from novices' perspective. In *14th International Conference on Evaluation and Assessment in Software Engineering (EASE)*, páginas 1–10.

Ros, R., Bjarnason, E., e Runeson, P. (2017). A machine learning approach for semi-automated search and selection in literature studies. In *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, páginas 118–127.

Salton, G. e Buckley, C. (1987). Term weighting approaches in automatic text retrieval. Relatório técnico, Cornell University.

Scells, H., Zuccon, G., e Koopman, B. (2019). Automatic boolean query refinement for systematic review literature search. In *The world wide web conference*, páginas 1646–1656.

Sundaram, G. e Berleant, D. (2023). Automating systematic literature reviews with natural language processing and text mining: A systematic literature review. In *International Congress on Information and Communication Technology*, páginas 73–92. Springer.

Thomas, J., McNaught, J., e Ananiadou, S. (2011). Applications of text mining within systematic reviews. *Research synthesis methods*, 2(1):1–14.

Turian, J., Ratinov, L., e Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, páginas 384–394.

van Dinter, R., Tekinerdogan, B., e Catal, C. (2021). Automation of systematic literature reviews: A systematic literature review. *Information and Software Technology*, 136:106589.

Van Rijsbergen, C. (1979). Information retrieval: theory and practice. In *Proceedings of the joint IBM/University of Newcastle upon tyne seminar on data base systems*, volume 79.

Vasconcellos, F. J., Landre, G. B., Cunha, J. A. O., Oliveira, J. L., Ferreira, R. A., e Vincenzi, A. M. (2017). Approaches to strategic alignment of software process improvement: A systematic literature review. *Journal of Systems and Software*, 123:45–63.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., e Polosukhin, I. (2017). Attention is all you need.

Weiss, S. M., Indurkhya, N., Zhang, T., e Damerau, F. (2010). *Text mining: predictive methods for analyzing unstructured information*. Springer Science & Business Media.

White, V. J., Glanville, J. M., Lefebvre, C., e Sheldon, T. A. (2001). A statistical approach to designing search filters to find systematic reviews: objectivity enhances accuracy. *Journal of information science*, 27(6):357–370.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, páginas 1–10.

Wohlin, C., Kalinowski, M., Felizardo, K. R., e Mendes, E. (2022). Successful combination of database search and snowballing for identification of primary studies in systematic literature studies. *Information and Software Technology*, 147:106908.

Zhang, H., Babar, M. A., e Tell, P. (2011). Identifying relevant studies in software engineering. *Information and Software Technology*, 53(6):625–637.

Zwakman, M., Verberne, L. M., Kars, M. C., Hooft, L., van Delden, J. J., e Spijker, R. (2018). Introducing palette: an iterative method for conducting a literature search for a review in palliative care. *BMC palliative care*, 17:1–9.