Análise do Problema de Overstemming Aplicado ao Contexto Matemático

Celso B. A. Patiri[1], **Pedro H. S. Tenório**[1], **Flavio B. Gonzaga**[1]

[1]Universidade Federal de Alfenas (UNIFAL-MG)
Alfenas – MG – Brasil

{a16001,a15034,fbgonzaga}@bcc.unifal-mg.edu.br

Abstract. *This article explores the problems caused by the stemming process in a search engine concerning mathematical content. Specifically, in cases where overstemming causes ambiguities between words of mathematical and non-mathematical context in the results of a query. A study is carried out in order to demonstrate occurrences of ambiguities and their causes, as well as the proposal of a listing of keywords that can be excluded from the stemming process in order to improve query results.*
keywords: *Stemming, Overstemming, Math Information Retrieval.*

Resumo. *Este artigo explora problemas causados pelo processo de stemização em uma ferramenta de busca para o conteúdo matemático na língua inglesa. Especificamente casos em que overstemming causa ambiguidades entre palavras de contexto matemático e não matemático nos resultados de uma busca. Um estudo é realizado a fim de demonstrar as ocorrências de ambiguidades e seus motivos, bem como a proposta de geração de uma lista de palavras-chave que podem ser excluídas do processo de stemização a fim de melhorar os resultados das buscas.*
palavras-chave: *Stemming, Overstemming, Recuperação de Informação Matemática.*

1. Introdução

A stemização é uma técnica desenvolvida no contexto de recuperação de informações com o intuito de normalizar os termos de alguma linguagem natural. O método consiste em reduzir o tamanho do vocabulário de modo que palavras com significados semelhantes sejam unidas em um termo em comum.

Para ilustrar esse conceito podemos considerar, por exemplo, o verbo "*to console*". Na língua inglesa existem diversas palavras como: "*consoled*", "*consoles*", "*consolation*", "*consoling*", e "*consolingly*" derivadas de "*console*". Assim, uma busca pela palavra "*consolation*" provavelmente estará interessada em documentos onde as outras palavras ocorram.

O processo de stemização é utilizado para se obter esse comportamento. A ideia é processar previamente as palavras de modo que uma dada palavra, assim como suas derivações e inflexões sejam reduzidas para um mesmo radical, para que assim, no processo de busca todas elas sejam mapeadas para o mesmo termo. É importante ressaltar que existem diferentes algoritmos de stemização (*stemmers*). Deste modo, é possível que uma

mesma palavra resulte em diferentes radicais após o processo de stemização, dependendo do algoritmo utilizado.

Considerando ainda o exemplo relacionado à palavra "*console*" e algumas possíveis variações, ao passar por um algoritmo de stemização, todas essas palavras poderiam ser reduzidas por exemplo para o radical "*consol*". Deste modo, com todas as palavras do conjunto sendo reduzidas para o mesmo radical, uma busca pela palavra "*consolation*" resultaria na busca do radical "*consol*", que por sua vez apresentaria as ocorrências das demais palavras desse conjunto.

De forma sucinta, a ideia da stemização é melhorar a precisão na recuperação de informações através da junção, em um único termo, de diferentes palavras que compartilham um significado comum [Harman 1991].

1.1. Erros em stemização

Apesar de resultar em uma melhoria significativa na qualidade dos resultados exibidos em ferramentas de busca textual, devido à natureza não exata das linguagens naturais, outros problemas e ambiguidades podem ser incitados por esse processo.

Dois tipos de erros principais são considerados no processo de stemização:

- *Overstemming* é quando duas palavras com significados diferentes são reduzidas para um mesmo radical. Tal ocorrência também é denominada falso-positivo.
- *Understemming* é quando duas palavras semelhantes, que deveriam ser reduzidas para o mesmo radical são reduzidas para radicais distintos. Esse erro também é denominado falso-negativo.

Paice [Paice 1994] descreve as métricas *Overstemming Index* (**OI**) e *Understemming Index* (**UI**), medidas em função dos erros de stemização em uma amostra de palavras. A relação dessas métricas é utilizada para calcular o *Stemming Weight* (**SW**). Em geral, um algoritmo que possua um *SW* maior, tenderá a reduzir mais as palavras em comparação a um algoritmo cujo *SW* seja menor.

Continuando o exemplo citado, podemos considerar palavras como "*consolatory*", e dependendo de como o algoritmo de stemização for implementado, se o parâmetro *SW* for pequeno demais, a palavra pode ser reduzida, por exemplo, para o radical "*consolator*" (*understemming*), gerando assim uma distinção entre o mapeamento desta palavra e das demais citadas, como "*console*", mesmo que para o contexto de buscas textuais essa equivalência seja desejada. Em contrapartida, se o *SW* do algoritmo de stemização for alto demais, uma palavra como "*consolidate*" poderia ser reduzida para o mesmo radical "*consol*" (*overstemming*), gerando assim ambiguidade no significado da busca.

Paice provou que com o aumento do *SW*, algoritmos de stemização aumentam ocorrências de *overstemming*, enquanto reduzem ocorrências de *understemming*. Em contrapartida, com a diminuição do parâmetro as ocorrências de *understemming* aumentam, enquanto as de *overstemming* diminuem. [Paice 1994].

O presente trabalho analisa o problema de *overstemming* aplicado no contexto de ferramentas de busca para o conteúdo matemático [SearchOnMath 2021] [ApproachZero 2021]. Assim, foram considerados inicialmente dois universos distintos: **i)** palavras de uso geral do idioma em inglês; e **ii)** palavras específicas do universo

matemático (também em inglês). O objetivo foi identificar palavras matemáticas que, ao serem buscadas, retornam também palavras fora do contexto matemático, em virtude do problema de *overstemming*.

Ao identificar palavras sem significado matemático, mas que seriam confundidas com termos matemáticos em virtude desse problema, foi possível a construção de uma lista de palavras que não devem passar pelo processo de stemização, a fim de minimizar tal problema. Um exemplo desse cenário pode ser visto nas palavras "*evening*" (uso geral) e "*even*" (termo matemático). Ambas podem resultar no mesmo radical ao passarem pelo processo de stemização (*even*), o que causaria resultados ambíguos caso um usuário buscasse por *even*.

Assim, os objetivos deste trabalho são:

1.2. Objetivo Geral

Identificar a parcela do vocabulário da base de dados suscetível a ambiguidades entre os contextos matemático e não matemático, e através dessa análise adequar o mecanismo de busca de modo a minimizar ocorrências de tais ambiguidades, melhorando assim a precisão e recuperação das informações.

1.3. Objetivos Específicos

- Obter um dicionário compreensivo da língua inglesa.
- Obter um dicionário compreensivo de termos matemáticos em inglês.
- Construir uma base de documentos matemáticos (retirados da Wikipedia [1] em inglês), a fim de se demonstrar o problema abordado.
- Analisar o problema de *overstemming* considerando ambos os dicionários.
- Gerar uma lista de palavras que não devem ser stemizadas a fim de minimizar o problema de *overstemming* no contexto matemático.
- Aplicar a lista de palavras na base de documentos matemáticos, de modo a demonstrar a redução do problema.

2. Trabalhos Relacionados

Problemas como *overstemming* e *understemming* são comuns na área de stemização, de forma generalizada, a questão entre equilíbrio da redução do vocabulário, e aumento de ambiguidades geradas por *overstemming* se faz presente em toda aplicação que utiliza a stemização. Essas questões são, em termos práticos, inerentes ao processo de stemização e seu propósito.

Essa seção procura expor alguns trabalhos recentes que abordam tais questões, trazendo à luz diferentes contextos em que os problemas do processo de stemização são encontrados e como estes são analisados.

O artigo [Abainia et al. 2017] apresenta a proposta de um *stemmer* para o idioma árabe, abordando os problemas de *understemming* e *overstemming* em vista às particularidades da morfologia da língua árabe. O autor cita irregularidades em regras de afixação, e como a adição ou remoção de afixos pode alterar radicalmente a semântica de uma palavra. Fazendo assim com que ocorrências de *overstemming* sejam mais críticas. Para

[1] `https://en.wikipedia.org/wiki/Main_Page`, acesso em 16/07/2021.

realizar um estudo comparativo, um conjunto de documentos foi coletado a partir de diversos fóruns de discussão, abrangendo diferentes variações modernas da língua árabe. O autor adotou a metodologia de Paice para avaliar os erros de stemização encontrados, de modo que os parâmetros (OI), (UI) e (SW) [Paice 1994] foram calculados tanto para o *stemmer* proposto quanto para outros *stemmers* clássicos existentes na literatura.

Outra proposta de *stemmer* é apresentada em [Kassim et al. 2016], para o idioma malaio. O autor explica como a maioria dos *stemmers* existentes para o idioma adotam uma abordagem baseada em regras para a remoção de afixos, e comumente referenciam dicionários externos para a resolução de conflitos na stemização. Devido à complexidades da morfologia do idioma, os algoritmos baseados em regras se fazem suscetíveis a diferentes tipos de erros de stemização. Esses erros foram classificados de acordo com o tipo de afixo, enumerados pelo autor na seguinte ordem: infixação; confixação; prefixação; e sufixação. Para aumentar a eficiência do algoritmo, é proposto que as regras de stemização referentes à redução de afixos sejam aplicadas na ordem acima citada. O autor conclui que ao estabelecer essa ordem, há uma minimização na ocorrência de erros, fazendo com que o algoritmo requeira um dicionário com menos entradas em comparação a outros *stemmers* do idioma.

3. Algoritmos de Stemização

Existem diversos algoritmos de stemização com diferentes abordagens. Os métodos se diferem em performance, complexidade, precisão, e maneira como os problemas da stemização são solucionados. Nesse sentido, pode-se classificar os algoritmos em duas categorias principais [Schofield and Mimno 2016]:

- Algoritmos baseados em regras: incluem métodos governados principalmente por um conjunto de regras para a conversão de um afixo para outro. Estes métodos são rápidos, porém limitados. A conversão de uma dada palavra **A** para um radical **B** é consistente independente de contexto, assim, estes algoritmos são determinísticos e não conseguem abranger exceções morfológicas e ambiguidades. A maioria dos métodos clássicos de stemização se enquadram nessa categoria, incluindo o *Porter Stemmer* [Porter 2001].
- Algoritmos baseados em contexto: têm como intuito contornar as imprecisões da primeira abordagem. Para isso, eles utilizam dicionários, análise das inflexões, e/ou análise da língua falada para determinar a forma reduzida adequada para cada palavra, baseado em seu contexto. Assim, ocorrências de uma mesma palavra **A** não serão necessariamente reduzidas para um mesmo radical **B** ao longo de um texto. [Schofield and Mimno 2016].

Devido a natureza inexata da área de stemização e linguagens naturais, a stemização está sempre ligada a ambiguidade [Moral et al. 2014]; nenhuma linguagem segue um conjunto determinístico de regras, e assim, não existe um *stemmer* perfeito, capaz de obter o radical correto de qualquer termo de forma sistemática.

No contexto da busca textual, é necessário que o algoritmo de stemização adotado seja aplicado tanto na base de documentos quanto nas consultas (submetidas posteriormente pelos usuários). A próxima seção apresenta mais detalhes sobre esse aspecto.

3.1. Stemização no contexto da busca textual

Conforme mencionado, o objetivo do presente trabalho é o de primeiramente analisar o problema de *oversteming* no contexto de buscadores textuais matemáticos, e em seguida, propor uma lista de palavras que não devem passar pelo processo de stemização a fim de minimizar tal problema.

No desenvolvimento de um buscador textual, o algoritmo de stemização é utilizado basicamente em dois momentos distintos: **i)** na construção do índice invertido; **ii)** em cada consulta submetida pelos usuários. Esse método garante a consistência no casamento entre os termos submetidos e aqueles previamente indexados na base de dados. Assim, por ser um algoritmo que será executado a cada consulta submetida, optou-se por analisar o problema de *oversteming* aplicado em algoritmos baseados em regras, que possuem uma menor complexidade computacional, e são, portanto, mais adequados ao uso em tempo real.

A seguir são apresentados três *stemmers* baseados em regras que são comumente utilizados: Paice-Husk (Lancaster Stemmer); Porter Stemmer; Porter2 Stemmer [Schofield and Mimno 2016].

3.1.1. Algoritmo de Paice-Husk

O algoritmo de Paice-Husk, ou Lancaster Stemmer utiliza um processo iterativo de remoção ou troca do sufixo das palavras, a partir de um conjunto externo de regras [Paice 1990]. Seu intuito é apresentar maior flexibilidade quando comparado com o Porter Stemmer, permitindo que um conjunto de regras customizáveis possa ser desenvolvido e implementado para aplicações específicas. O algoritmo é comumente considerado como tendo um SW alto, apresentando assim grandes ocorrências de *overstemming* [Jivani et al. 2011].

3.1.2. Porter Stemmer & Snowball

O Porter Stemmer [Porter 1980] é um dos mais antigos algoritmos de stemização, e até hoje o mais comumente utilizado. Ele consiste em cinco fases de regras e condições que verificam padrões em sequências de vogais e consoantes, reduzindo sufixos na língua inglesa. O algoritmo é reconhecido por sua velocidade e simplicidade. As raízes produzidas muitas vezes não são palavras reais, como por exemplo a palavra "*stay*" é reduzida para o termo "*stai*".

Snowball é uma linguagem de programação de processamento de strings, desenvolvida para a criação de algoritmos de stemização baseados em regras [Porter 2001]. Nela, regras de stemização são definidas para que possam ser aplicadas de modo geral em diferentes vocabulários, possibilitando assim o desenvolvimento de algoritmos para outros idiomas (além da língua inglesa).

Em 2001, Porter publicou o desenvolvimento de um novo algoritmo, apresentado como uma revisão do *Porter Stemmer*. Este é muitas vezes referido como *Porter2* ou *Snowball Stemmer*. Segundo [Porter 2001]:

> "...após vinte anos de uso do *Porter Stemmer*, certas melhorias apresentaram-se, e assim, um novo algoritmo de stemização é aqui oferecido. Este pode ser chamado de '*Porter2*' *stemmer* para se distinguir do *Porter Stemmer*, do qual ele deriva."

Este novo algoritmo é, geralmente, considerado como sendo uma versão melhorada do *Porter Stemmer* original. As mudanças apresentam alterações nas fases originais do algoritmo anterior, bem como no tratamento de certos sufixos, exceções e apóstrofes. Assim, em virtude do *Porter2* ser um *stemmer* comumente utilizado, além de ser o algoritmo padrão do *Elasticsearch* [2] (a ser mais detalhado na seção de Metodologia); o mesmo foi o escolhido para este trabalho.

4. Metodologia e Resultados

Com o objetivo de verificar a ocorrência de *overstemming* entre palavras gerais do idioma inglês, e palavras específicas do contexto matemático, foi necessário o uso de dois dicionários:

- Idioma inglês;
- Termos matemáticos (em inglês).

4.1. Seleção dos dicionários

Para o idioma inglês, o dicionário utilizado foi o *Collins Online English Dictionary* [Collins 2018]; enquanto que para o dicionário de termos matemáticos (em inglês), o escolhido foi: *The Concise Oxford Dictionary of Mathematics* [Clapham et al. 2014]. Ambos são dicionários produzidos por editoras mundialmente conceituadas, aspecto determinante considerado pelos autores no momento da seleção dos dicionários a serem utilizados.

4.2. Normalização dos dicionários

A partir dos dicionários, foram geradas duas listas iniciais formadas por todas as palavras contidas em cada um. A *lista_inglês*, obtida a partir do dicionário *Collins*, continha $238\,422$ entradas, enquanto que a *lista_matemática*, obtida a partir do dicionário matemático *Oxford*, continha $2\,272$ entradas.

A *lista_inglês* no entanto, precisou passar por mais alguns tratamentos. Observou-se a existência de entradas complexas que não representam palavras de uso geral na língua inglesa, como por exemplo: componentes químicos; expressões contendo números, caracteres especiais ou acrônimos; ou termos compostos de múltiplas palavras (e.g., "*3-hydroxybutanal*", "*A & M college*", "*guest bedroom*").

Assim, retirou-se da *lista_inglês* entradas contendo caracteres especiais. Entradas contendo mais de uma palavra foram divididas em termos individuais. Após a realização desses tratamentos, o número de entradas únicas resultante na *lista_inglês* foi de $144\,152$.

Por fim, como o objetivo do trabalho é a detecção de *overstemming* entre palavras de uso geral do idioma inglês, e termos matemáticos, retirou-se da *lista_inglês* as

[2] `https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-stemmer-tokenfilter.html`, acesso em 30/07/2021.

palavras que estivessem na *lista_matemática*. Assim, se uma palavra da *lista_inglês* resultar no mesmo radical de uma palavra da *lista_matemática*, garante-se que são palavras diferentes, e que possivelmente possa se tratar de um caso de *overstemming*.

Após a remoção da *lista_matemática* da *lista_inglês*, o número total de entradas na *lista_inglês* foi de 142 186.

As listas foram então submetidas ao algoritmo *Snowball*, de modo que o radical de cada palavra, resultante do processo de stemização, foi obtido. Assim, a quantidade de radicais (sem repetição) extraída da *lista_inglês* foi de 112 083; enquanto que a quantidade de radicais (sem repetição) extraída da *lista_matemática* foi de 1 947.

4.3. Análise de *overstemming*

Com os dicionários normalizados, e as respectivas listas de radicais das palavras obtidas, inicia-se então a etapa de análise quanto à possível ocorrência de *overstemming*. Nesse sentido, tomando como base os radicais das palavras (presentes em ambos os dicionários), os radicais foram organizados em três conjuntos:

1. radicais cujas palavras estão presentes apenas na *lista_inglês*;
2. radicais cujas palavras estão presentes apenas na *lista_matemática*;
3. radicais cujas palavras estão presentes em ambas as listas (*lista_inglês* e *lista_matemática*), sendo portanto a interseção entre os conjuntos 1 e 2.

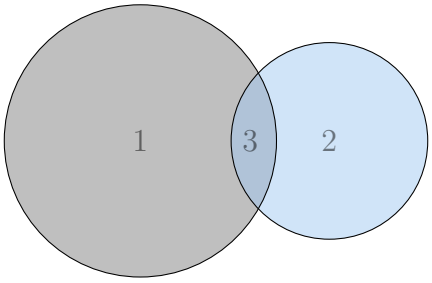

Figura 1. Diagrama de interseção entre os conjuntos de radicais.

Como o objetivo do trabalho é minimizar o problema de *overstemming* entre palavras que não são do contexto matemático, em relação a palavras matemáticas, o conjunto 3 é o foco do desenvolvimento. Assim, na montagem do índice invertido, todas as palavras relacionadas aos radicais presentes nos grupos 1 e 2 passam pelo processo de stemização normalmente. Palavras vinculadas aos radicais presentes no grupo 3, que são oriundas de ambos os contextos (matemático e não matemático) poderão ou não passar pelo processo de stemização.

Aprofundando mais no conjunto 3, ainda é possível separá-lo em dois subconjuntos no que diz respeito aos termos matemáticos. Os subconjuntos são descritos a seguir:

3.1 Termos matemáticos que, após passar pelo processo de stemização, não são modificados (e.g., termo: *even*; radical: *even*). Estão presentes 471 termos neste subconjunto.
3.2 Termos matemáticos que, após passar pelo processo de stemização, resultam em um radical diferente do termo (e.g., termo: *objective*: radical: *object*). Estão presentes 876 termos neste subconjunto.

Os subconjuntos acima definidos são importantes porque eles guiam as ações no sentido de submeter ou não uma palavra ou termo matemático ao processo de stemização. Para o subconjunto **3.1**, como o termo é igual ao seu radical, caso se detecte a ocorrência de *overstemming* com alguma palavra não matemática, a única opção possível é a de não realizar a stemização na palavra. Por exemplo, considere a seguinte relação entre o radical *even*, o termo matemático *even* e a palavra em inglês *evening*. Permitir que todos passem pelo processo de stemização fará com que *evening* seja reduzido para *even*. Com isso, tanto buscas pelo termo matemático *even* retornarão também resultados contendo *evening*, como buscas pela palavra *evening* retornarão resultados contendo o termo matemático *even*. Este é um exemplo de cenário de *overstemming*, e que pode ser interessante evitar. A única ação possível seria não permitir que *evening* fosse stemizada (na criação do índice invertido, e posteriormente nas consultas), o que resolveria o problema de *overstemming* para este caso. A Figura 2 mostra um exemplo para este problema no buscador *SearchOnMath*, onde múltiplos resultados contendo *even* são retornados para uma busca pela palavra *evening*.

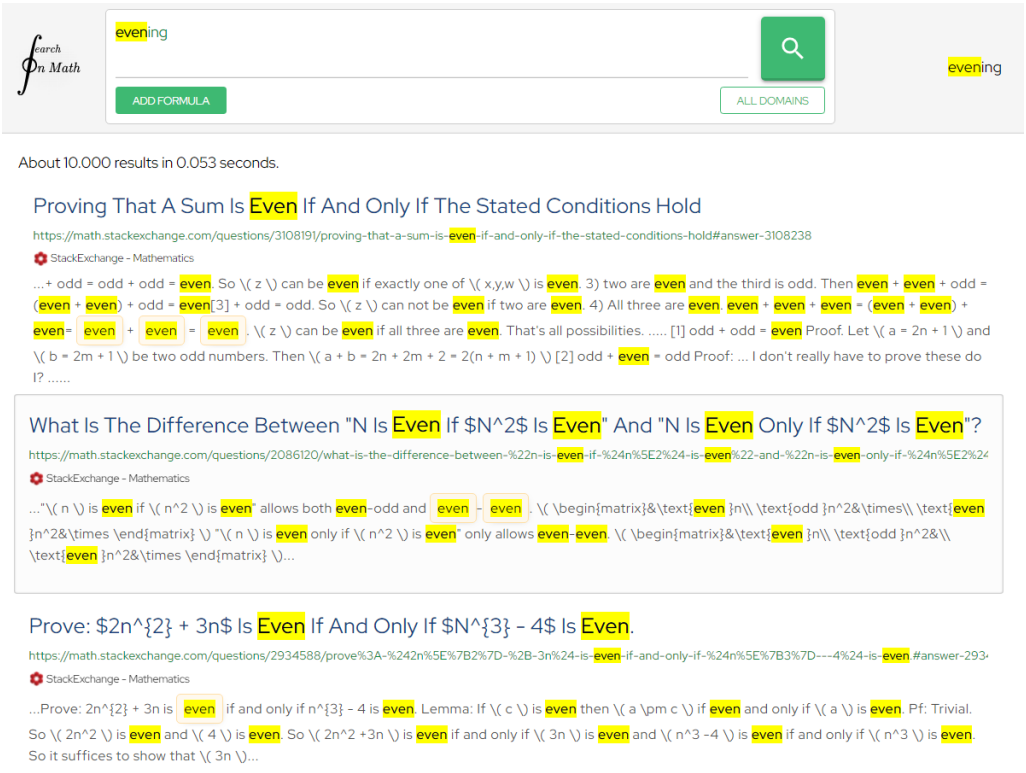

Figura 2. Exemplo de busca pela palavra "*evening*" na *SearchOnMath*.

Considerando-se o subconjunto **3.2**, como os termos matemáticso são diferentes dos respectivos radicais, tem-se a possibilidade de stemizar ou não tanto os termos matemáticos, quanto, possivelmente, as palavras. Neste cenário, deve-se analisar caso a caso de modo a primeiro identificar se há ou não *overstemming*, e em seguida, decidir qual ação tomar.

4.4. Listagem final

Uma vez definidos os conjuntos apresentados na seção anterior, iniciou-se o processo de construir uma lista final de palavras e termos matemáticos que não deverão ser stemizados, a fim de se minimizar o problema de *overstemming* entre estes dois universos.

4.4.1. Palavras não matemáticas com radical matemático

Nesta seção são analisadas as palavras não matemáticas, mas que uma vez stemizadas, resultam em um radical contido no subconjunto **3.1**. Ou seja, o radical da palavra não matemática é igual a um termo matemático.

Neste cenário, identificam-se duas possibilidades, cada uma tendo vantagens e desvantagens. A questão a ser analisada é que na lista de palavras não matemáticas, múltiplas palavras semelhantes podem dividir o mesmo radical. A primeira possibilidade, portanto, seria acrescentar tais palavras do dicionário à lista de palavras que não devem ser stemizadas. Esta ação resolveria o problema da ambiguidade com o termo matemático, mas acrescentaria o problema de perder a relação de semelhança entre elas (que são semelhantes em significado). Ao escolher essa possibilidade, 145 palavras do universo não matemático seriam excluídas do processo de stemização.

A segunda possibilidade seria o caminho contrário, ou seja, não acrescentar as palavras do dicionário à lista de palavras que não devem ser stemizadas. Esta ação não resolveria o problema da ambiguidade com o termo matemático, mas acrescentaria a capacidade de se retornar resultados considerando-se a relação de semelhança entre as palavras do dicionário não matemático.

No trabalho optou-se pela segunda possibilidade. Entende-se que a relação de semelhança entre as palavras é um ganho maior do que o dano causado pela ambiguidade das palavras com o termo matemático, em virtude disso, nenhuma palavra do universo não matemático, cujo radical seja igual a um termo matemático, foi adicionada na lista de palavras excluídas do processo de stemização. Ainda nesse cenário, intuitivamente falando, espera-se que uma consulta (*query*) seja composta não apenas por uma, mas por múltiplas palavras. Essa característica também ajudaria a minimizar os danos de *overstemming* em apenas uma das palavras da consulta.

4.4.2. Palavras não matemáticas e termos matemáticos com radical em comum

Nesta seção são analisadas as palavras não matemáticas e termos matemáticos que, uma vez stemizados, resultam em um mesmo radical. Caso esse dos termos matemáticos contidos no subconjunto **3.2**.

Neste cenário, quando se identificou que um determinado termo matemático possui significado diferente das palavras, o termo foi acrescentado à lista de palavras que não devem ser stemizadas, para evitar ambiguidade. Como por exemplo: o termo "*factorial*" compartilha o radical "*factori*" com as palavras não matemáticas "*factory*" e "*factories*". A fim de diferenciar o mapeamento do termo com as palavras em uma busca, o termo é adicionado à lista, removendo-o assim do processo de stemização.

No entanto, o termo pode dividir o radical com palavras de significado semelhante, assim como palavras de significado divergente. Dessa forma, quando excluído do processo de stemização, o termo perde a relação de equivalência com as palavras semelhantes, o que pode reduzir a quantidade de resultados relevantes na busca. Para lidar com esse problema, cada caso foi analisado para definir se é preferível excluir o termo do processo de stemização, perdendo assim a equivalência no mapeamento com palavras semelhantes, ou mantê-lo no processo de stemização, gerando assim ambiguidade com o radical de uma ou mais palavras não matemáticas.

Tomando como exemplo os termos "*interpolating*" e "*interpolation*", ambos dividem o radical "*interpol*", que é por sua vez uma palavra não matemática. Dessa forma, os dois termos podem ser adicionados a lista de forma a eliminar as ambiguidades com a palavra "*interpol*". Porém, a remoção dos termos do processo de stemização remove também a equivalência de mapeamento entre eles, fazendo com que a busca por um não apresente resultados contendo o outro. Nesse caso, portanto, foi definido que stemizar normalmente os termos, mantendo assim a relação entre eles, traz mais benefícios do que eliminar as ambiguidades com o termo "*interpol*".

Outro exemplo é o termo "*objective*", que quando stemizado resulta no radical "*object*", que por sua vez é também o radical das palavras não matemáticas: "*object*", "*objection*", "*objections*", "*objectives*" e "*objectivity*". Dentre elas, as três primeiras não possuem significado semelhante ao termo "*objective*", enquanto as outras duas são de fato semelhantes. Dessa forma, a exclusão do termo do processo de stemização causa a remoção de ambiguidades, assim como a perda de equivalências desejadas. Diante desse cenário, permitimos que o termo seja stemizado normalmente, visando que manter a equivalência entre "*objective*", "*objectives*", e "*objectivity*" é uma troca desejável ao custo das ambiguidades.

Após realizar essa análise caso a caso, seguindo os critérios descritos nesta seção, obteve-se uma lista final contendo 35 termos matemáticos que não devem ser stemizados, a fim de se minimizar o problema de *overstemming*. A lista está apresentada a seguir na Tabela 1.

acute	chaos	factorial	machine	response
amicable	compasses	forbes	many	rolle
baseline	concentric	hawking	marie	runge
basis	conjugate	hawthorne	monte	series
bayes	decomposition	hooke	orientable	universal
boole	denis	insolvable	places	vertices
casting	entering	instance	principal	viete

Tabela 1. Lista de palavras que não devem ser stemizadas

4.5. Implementação e Experimentos Práticos

O próximo passo foi testar os efeitos da aplicação da lista gerada. Seguindo o contexto de recuperação de informações, um corpo de documentos foi coletado como base para a realização de buscas (*queries*), e então tratado de modo a possibilitar a verificação dos resultados retornados com e sem a aplicação da lista. Devido a natureza do trabalho,

foi preferível escolher uma base de dados consistente com o contexto matemático. Para isso, os documentos foram obtidos a partir de páginas pertencentes à *Wikipedia* em Inglês. As páginas, obtidas em Janeiro de 2021, são aquelas que continham pelo menos uma fórmula matemática. Cada página foi armazenada em um documento na base de dados, definido com os campos: "título", "URL", e "conteúdo"; totalizando $36\,762$ documentos.

Para o processo de armazenamento e busca dos dados, a ferramenta *Elasticsearch* foi escolhida. O *Elasticsearch* [3] é um mecanismo de busca e análise de dados distribuído, gratuito e aberto para diversos tipos de dados. Conhecido por suas REST APIs simples, natureza distribuída, velocidade e escalabilidade, o *Elasticsearch* é o componente central do *Elastic Stack* (ou *ELK Stack*), um conjunto de ferramentas com amplas funcionalidades na gestão e busca de dados. Para os propósitos deste trabalho, no entanto, as funcionalidades de armazenamento e busca de dados textuais serão o foco principal.

Dentro da ferramenta, toda unidade de dados armazenada é definida como um documento (*document*), na forma de um objeto JSON. Uma coleção de documentos que possuem características semelhantes é chamada de índice (*index*). Durante o processo de inserção de um documento em um índice (indexação), ao lidar com valores textuais, a ferramenta pode executar um processo de análise (*analysis* ou *text analysis*), onde o texto é analisado, tratado e armazenado como uma estrutura de dados eficiente para a busca. Além disso, o comportamento da análise pode ser configurado em um analisador (*analyzer*), definindo quais mecanismos vão ser utilizados durante o processo.

Um analisador é composto por três etapas: *character filters*, *tokenizer* e *token filters*. Onde *character filters* são responsáveis por adicionar, remover ou modificar caracteres do texto analisado (neste trabalho o uso de um *character filter* não se fez necessário); o *tokenizer* é responsável por dividir o texto em *tokens*, removendo pontuações e separando-o por espaços, hifens, etc; e por fim, os *token filters* atuam de forma semelhante aos *character filters*, porém sobre os *tokens* gerados na etapa anterior. Vale notar ainda, que *token filters* são aplicados na ordem em que são especificados. Para manter a consistência da busca de valores textuais com os documentos indexados, o mesmo analisador utilizado em um dado índice é também aplicado em cada busca.

Neste trabalho, dois índices foram organizados com o intuito de diferenciar as buscas sem e com a aplicação da lista gerada (*indice_sem_lista* e *indice_com_lista*, respectivamente), ambos utilizando analisadores customizados, aplicando os seguinte *token filters* (em ordem):

- **ASCII folding** [4]: Converte todo carácter alfabético, numérico ou simbólico que não pertence ao bloco *Basic Latin Unicode* (primeiros 127 caracteres ASCII) ao seu equivalente ASCII, se este existir. O filtro troca, por exemplo: o carácter "à" por "a".
- **Lowercase** [5]: Converte todo carácter alfabético maiúsculo por seu equivalente minúsculo.

[3] `https://www.elastic.co/what-is/elasticsearch`, acesso em 10/08/2021.

[4] `https://www.elastic.co/guide/en/elasticsearch/reference/7.9/analysis-asciifolding-tokenfilter.html`, acesso em 25/08/2021.

[5] `https://www.elastic.co/guide/en/elasticsearch/reference/7.9/analysis-lowercase-tokenfilter.html`, acesso em 25/08/2021.

- **Snowball** [6]: Submete todo *token* a um algoritmo de stemização baseado na linguagem *Snowball*. Esse filtro possui suporte para diversos idiomas.

No *indice_com_lista*, para excluir do processo de stemização a lista gerada, foi necessário utilizar um *token filter* adicional. O filtro escolhido, ***keyword_marker*** [7], marca os *tokens* especificados de forma que estes são ignorados por filtros de stemização subsequentes. No caso desse índice, este filtro é aplicado antes do *Snowball*.

As Figuras 3(a) e 3(b) apresentam os analisadores customizados que foram utilizados:

```
"analysis": {
  "analyzer": {
    "analisador_sem_lista": {
      "filter": [
        "asciifolding",
        "lowercase",
        "snowball"
      ],
      "char_filter": [],
      "type": "custom",
      "tokenizer": "standard"
    }
  }
}
```

(a) indice_sem_lista

```
"analysis": {
  "filter": {
    "nao_stemizar": {
      "type": "keyword_marker",
      "keywords_path": "C:\\..."
    }
  },
  "analyzer": {
    "analisador_com_lista": {
      "filter": [
        "asciifolding",
        "lowercase",
        "nao_stemizar",
        "snowball"
      ],
      "char_filter": [],
      "type": "custom",
      "tokenizer": "standard"
    }
  }
}
```

(b) indice_com_lista

Figura 3. Analisadores customizados utilizados nos índices

Na Figura 3(b), o filtro customizado *nao_stemizar* é definido. Este é o *token filter* utilizado para excluir as palavras da lista gerada do processo de stemização. O parâmetro "*type*" define que esse filtro é do tipo *keyword_marker*, e o parâmetro "*keywords_path*" define a localização da lista no computador, para que as palavras sejam carregadas.

Após a indexação da base de dados nos índices, a fim de demonstrar as diferenças nos resultados, a escolha das palavras a serem buscadas foi feita a partir da lista gerada, tendo em vista que as palavras dessa lista estão relacionadas a casos de ambiguidade, e intuitivamente falando, devem apresentar resultados diferentes de acordo com o índice utilizado.

Na ferramenta *Elasticsearch*, as buscas realizadas geralmente retornam uma variedade de informações relacionadas aos documentos e ao índice, o que pode dificultar a visualização. Para apresentar os efeitos nos resultados de forma mais sucinta optou-se por exibir, para cada documento, apenas seu valor de ranqueamento, título, e alguns fragmentos do conteúdo em que a palavra buscada é encontrada.

[6] `https://www.elastic.co/guide/en/elasticsearch/reference/7.9/analysis-snowball-tokenfilter.html`, acesso em 25/08/2021.

[7] `https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-keyword-marker-tokenfilter.html`, acesso em 25/08/2021.

Para ilustrar os resultados, foi decidido por utilizar os termos matemáticos: "*concentric*", que apresenta ambiguidade com palavras como "*concentrated*" e "*concentration*"; e também o termo "*vertices*", que apresenta ambiguidade com as palavras "*vertically*" e "*verticalness*".

Os resultados das buscas desses termos em ambos os índices estão apresentados na Figuras 4 e 5:


```
hits:
  total:
    value: 2989
  hits:
  - _score: 5.420294
    _source:
      title: "Concentration ratio"
    highlight:
      content:
      - "Concentration ratio In"
      - ", concentration ratios are used"
  - _score: 5.3873663
    _source:
      title: "Air pollutant concentrations"
    highlight:
      content:
      - " concentrations Air pollutant"
      - " concentrations, as measured or"
  - _score: 5.3860655
    _source:
      title: "Concentration"
    highlight:
      content:
      - "Concentration In"
      - " chemistry, concentration is the"
```


(a) Busca no indice_sem_lista


```
hits:
  total:
    value: 290
  hits:
  - _score: 9.464475
    _source:
      title: "Concentric tube heat exchanger"
    highlight:
      content:
      - "Concentric tube heat"
      - " exchanger Concentric Tube"
  - _score: 9.462759
    _source:
      title: "Ventricular hypertrophy"
    highlight:
      content:
      - " categories: concentric"
      - ") hypertrophy. Concentric"
  - _score: 9.192632
    _source:
      title: "Steiner chain"
    highlight:
      content:
      - " concentric circles; in"
      - " given circles are concentric"
```


(b) Busca no indice_com_lista

Figura 4. Resultados das buscas pela palavra "*concentric*" em ambos os índices


```
hits:
  total:
    value: 4450
  hits:
  - _score: 4.545558
    _source:
      title: "Hypercube"
    highlight:
      content:
      - " vertices) are the 2n"
      - " of vertices (points) of"
  - _score: 4.5383563
    _source:
      title: "Vertical tangent"
    highlight:
      content:
      - "Vertical tangent"
      - " Vertical tangent on the function"
  - _score: 4.532288
    _source:
      title: "Vertical exaggeration"
    highlight:
      content:
      - "Vertical exaggeration A"
      - " vertically exaggerated"
```


(a) Busca no indice_sem_lista


```
hits:
  total:
    value: 2115
  hits:
  - _score: 6.1445804
    _source:
      title: "Hypercube"
    highlight:
      content:
      - " vertices) are the 2n"
      - " of vertices (points) of"
  - _score: 6.081056
    _source:
      title: "Vertex (geometry)"
    highlight:
      content:
      - " plural form: vertices or"
      - " vertices.[2][3][4] Definition"
  - _score: 6.0772867
    _source:
      title: "Cage (graph theory)"
    highlight:
      content:
      - " few vertices as possible"
      - " number of vertices, among"
```


(b) Busca no indice_com_lista

Figura 5. Resultados das buscas pela palavra "*vertices*" em ambos os índices

Nota-se que as buscas realizadas no *indice_sem_lista* apresentam um número maior de documentos retornados comparado com o *indice_com_lista*. No caso da busca pelo termo "*concentric*", os números foram $2\,989$ e 290. Enquanto na busca por "*vertices*", $4\,450$ e $2\,115$. Além disso, a aplicação da lista apresenta uma melhora na pontuação ("*score*") dos primeiros documentos retornados pelas buscas.

5. Conclusões

Considerando os resultados encontrados, é possível identificar diferenças significativas ao remover as palavras da lista do processo de stemização. Para analisar a melhora nas pontuações dos documentos, é valido citar o algoritmo de ranqueamento Okapi BM25 [Robertson and Zaragoza 2009], utilizado pelo *Elasticsearch* . O algoritmo normaliza a pontuação de um documento para uma dada busca, em função da raridade dos termos buscados. Em outras palavras, quanto menos frequente é um termo na coleção de documentos, maior o valor que ele contribui no ranqueamento de documentos que o contêm.

De maneira intuitiva, entende-se que ao remover um termo da lista do processo de stemização, ele deixa de ser generalizado de forma equivalente a outras palavras, reduzindo assim o número de documentos em que o termo é encontrado. Essa redução, além de evitar as possíveis ambiguidades, faz com que o termo excluído se torne mais valioso para a busca.

Devido a natureza ambígua do processo de stemização e da língua inglesa, a remoção das palavras não é um método exato. Esse processo pode causar erros imprevistos, ou remover equivalências desejáveis (considerando ainda que a *lista_matematica* não contém necessariamente todo termo matemático possível na língua inglesa). No entanto, concluise que no contexto de buscas matemáticas da *SearchOnMath*, a aplicação da lista gerada contribui para a obtenção de resultados relevantes.

5.1. Trabalhos Futuros

Como discutido ao longo do trabalho, quando lidando com linguagens naturais, não é possível apresentar métodos exatos ou ótimos para um estudo de erros de stemização. Neste trabalho foi decidido excluir apenas termos matemáticos do processo de stemização, considerando que estes são os mais significativos no contexto estudado. Essa decisão, no entanto, ignora um conjunto de ambiguidades em potencial (como elaborado na seção 4.4.1).

É possível repetir os experimentos deste trabalho, considerando também remover do processo de stemização palavras não matemáticas. Uma análise pode ser realizada para cada caso de ambiguidade encontrado, estudando os efeitos positivos e negativos da remoção das palavras.

Através da geração de uma nova lista, dois índices podem ser gerados para comparar os efeitos da remoção de termos matemáticos do processo de stemização, com a remoção de termos matemáticos juntamente com palavras não matemáticas. Assim, seria possível realizar por exemplo testes com usuários, verificando-se através de *feedback*, qual dos índices apresenta resultados mais adequados.

Referências

Abainia, K., Ouamour, S., and Sayoud, H. (2017). A novel robust arabic light stemmer. *Journal of Experimental & Theoretical Artificial Intelligence*, 29(3):557–573.

ApproachZero (2021). Approach zero: A math-aware search engine. `https://approach0.xyz/search/`. [Online; accessed 19-July-2021].

Clapham, C., Nicholson, J., and Nicholson, J. R. (2014). *The concise Oxford dictionary of mathematics*. Oxford University Press.

Collins (2018). Collins english dictionary. 13ª edição. 2018. `https://www.collinsdictionary.com/dictionary/english`. Accessed: 2021-06-24.

Harman, D. (1991). How effective is suffixing? *Journal of the american society for information science*, 42(1):7–15.

Jivani, A. G. et al. (2011). A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, 2(6):1930–1938.

Kassim, M. N., Maarof, M. A., Zainal, A., and Wahab, A. A. (2016). Enhanced affixation word stemmer with stemming error reducer to solve affxation stemming errors. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 8(3):37–41.

Moral, C., de Antonio, A., Imbert, R., and Ramírez, J. (2014). A survey of stemming algorithms in information retrieval. *Information Research: An International Electronic Journal*, 19(1):n1.

Paice, C. D. (1990). Another stemmer. In *ACM Sigir Forum*, volume 24, pages 56–61. ACM New York, NY, USA.

Paice, C. D. (1994). An evaluation method for stemming algorithms. In *SIGIR'94*, pages 42–50. Springer.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*.

Porter, M. F. (2001). Snowball: A language for stemming algorithms.

Robertson, S. and Zaragoza, H. (2009). *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Schofield, A. and Mimno, D. (2016). Comparing apples to apple: The effects of stemmers on topic models. *Transactions of the Association for Computational Linguistics*, 4:287–300.

SearchOnMath (2021). Searchonmath: A powerful search engine for math formulas. `https://www.searchonmath.com`. [Online; accessed 19-July-2021].