UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Mailton Fernandes de Carvalho

Um processo para construção de Tesauros de Domínio Específico no Contexto de uma Empresa de Teste de Software

Recife

2021

Mailton Fernandes de Carvalho

Um processo para construção de Tesauros de Domínio Específico no Contexto de uma Empresa de Teste de Software

Trabalho apresentado ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de Mestre em Ciência da Computação. Área de concentração: Inteligência Computacional

Orientador: Ricardo Bastos Cavalcante Prudencio

Coorientadora: Flávia de Almeida Barros

Recife

2021

Catalogação na fonte
Bibliotecária Nataly Soares Leite Moro, CRB4-1722

C331p Carvalho, Mailton Fernandes de
Um processo para construção de tesauros de domínio específico no contexto de uma empresa de teste de software / Mailton Fernandes de Carvalho. – 2021.
88 f.: il., fig., tab.

Orientador: Ricardo Bastos Cavalcante Prudêncio.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2021.
Inclui referências.

1. Inteligência computacional. 2. Criação de tesauros. 3. Recuperação de informação. 4. Mineração de texto. 5. Teste de software. I. Prudêncio, Ricardo Bastos Cavalcante (orientador). II. Título

006.31 CDD (23. ed.) UFPE - CCEN 2022 – 24

Mailton Fernandes de Carvalho

Um processo para construção de Tesauros de Domínio Específico no Contexto de uma Empresa de Teste de Software

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Inteligência Computacional

Aprovado em: 25/11/2021.

_____

Orientador: Prof. Dr. Ricardo Bastos Cavalcante Prudêncio

BANCA EXAMINADORA

_____

Prof. Dr. Alexandre Cabral Mota
Centro de Informática / UFPE

_____

Prof. Dr. Rafael Ferreira Leite de Mello
Departamento de Computação / UFRPE

_____

Profa. Dra. Flavia de Almeida Barros
Centro de Informática/ UFPE
(Co-orientadora)

AGRADECIMENTOS

Primeiramente, agradeço a Deus por sempre estar ao meu lado nos momentos mais difíceis, me guiando e iluminando os caminhos para ultrapassar todos os obstáculos ao longo dessa jornada. Obrigado, Senhor!

Aos meus pais, Maria José Fernandes de Carvalho e Antonio madruga de Carvalho pelo apoio e amor incodicional, pelo exemplo de vida, por acreditar e por todo o incentivo dado a mim para que eu pudesse alcançar objetivos que jamais imaginei que poderia alcançar.

A minha esposa, Joice Rafaela de Araújo Fernandes, pelo companheirismo, incentivo, paciência, apoio incodicional e por me fazer acreditar que sou capaz, nunca deixando de confiar em meu potencial.

Ao meu Filho Bento, que neste momento que estou escrevendo está na barriga da mamãe, faltando menos de um mês para nascer, eu e a mamãe estamos ansiosos para lhe conhecer, filho.

Aos meus irmãos, que de maneira direta e indireta contribuíram com palavras de conforto e incentivos.

Aos meus amigos, em especial, Antonio Braz Silva Finizola, Rodrigo Emerson Valentim da Silva, Cloves Alberto Chaves de Lima pelo apoio, incentivo e por compartilhar angústias e momentos descontraídos.

Aos professores. Drº Ricardo Prudêncio, Drº Alexandre Mota e em especial à professora Drª Flávia de Almeida Barros, por toda sua atenção, direcionamento e orientações, que sempre me guiou para o melhor caminho, tanto na vida acadêmica, profissional e pessoal. Seu papel foi de extrema importância nesta etapa de minha vida. Obrigado, Professores.

RESUMO

Atualmente, grandes empresas de software mantêm repositórios com milhares de documentos textuais. Contudo, as variações de terminologia verificadas nos documentos internos dificultam o *processamento automático dos textos*, tão importante nesses contextos. Esse problema se intensifica no cenário atual, no qual empresas usam sistemas de Recuperação de Informação (RI) para localizar documentos relevantes para uma diversidade de tarefas. Como os termos usados nas consultas nem sempre estão presentes nos documentos relevantes, nem sempre é possível recuperar todos os documentos relevantes para a atividade em curso. Uma opção para melhorar o desempenho de sistemas de processamento de texto é a utilização de Tesauros (dicionários de sinônimos e termos relacionados). Tesauros podem ser usados tanto em sistemas de RI, para expansão automática de consultas, como para melhorar o desempenho de classificadores de texto, ou apenas para prover um vocabulário padrão para composição de documentos internos da empresa. Contudo, dicionários genéricos, como o WordNet, não são apropriados para aplicações de domínio específico, principalmente devido à ambiguidade lexical. Nesse contexto, este trabalho propõe um processo para a geração de tesauros de domínio específico, criados a partir de documentos disponíveis nos repositórios das empresas. O processo recebe como entrada documentos textuais, e extrai desses documentos os termos relevantes para o domínio da Empresa. Em seguida, os termos relacionados são agregados em conjuntos. O processo utiliza técnicas de Processamento de Linguagem Natural, como POS-tagging, RegEx e n-gramas para enriquecer o dicionário de sinônimos com termos compostos, siglas e abreviações. Finalmente, os conjuntos de saída de palavras/termos relacionados são indexados e disponibilizados para consulta manual ou automática. Este trabalho foi desenvolvido no contexto de um projeto de cooperação de pesquisa em Teste de SW entre a Motorola Mobility (uma empresa Lenovo) e o CIn-UFPE. Como tal, nosso estudo de caso se concentrou no domínio de teste de SW. O tesauro construído através do protótipo implementado foi na expansão automática de consultas para melhorar o desempenho de dois sistemas de RI da Motorola Mobility. Testes comparativos revelaram melhores taxas de cobertura de recuperação e melhorias na classificação de documentos relevantes nesses dois sistemas. Esses resultados são muito importantes em cenários do mundo real, nos quais os testadores nem sempre têm tempo para analisar todos os documentos existentes.

Palavras-chave: criação de tesauros; recuperação de informação; expansão automática de consultas; mineração de texto; teste de software.

ABSTRACT

Currently, large software companies maintain repositories with thousands of textual documents. However, the variations in terminology found in internal documents make the automatic processing of texts, which is so important in these contexts, difficult. This problem is intensified in the current scenario, in which companies use Information Retrieval (IR) systems to locate relevant documents for a variety of tasks. As the terms used in queries are not always present in the relevant documents, it is not always possible to retrieve all documents relevant to the ongoing activity. One option to improve the performance of word processing systems is the use of Thesaurus (dictionaries of synonyms and related terms). Thesaurus can be used either in IR systems, for automatic query expansion, or to improve the performance of text classifiers, or just to provide a standard vocabulary for composing internal company documents. However, generic dictionaries such as WordNet are not suitable for domain specific applications, mainly due to lexical ambiguity. In this context, this work proposes a process for the generation of domain-specific thesauruses, created from documents available in companies' repositories. The process receives textual documents as input, and extracts from these documents the terms relevant to the Company's domain. The related terms are then aggregated into sets. The process uses Natural Language Processing techniques such as POS-tagging, RegEx and n-grams to enrich the thesaurus with compound terms, acronyms and abbreviations. Finally, the output sets of related words/terms are indexed and made available for manual or automatic query. This work was developed in the context of a research cooperation project on SW Testing between Motorola Mobility (a Lenovo company) and CIn-UFPE. As such, our case study focused on the SW testing domain. The thesaurus built through the implemented prototype was the automatic expansion of queries to improve the performance of two RI systems from Motorola Mobility. Comparative testing has revealed better retrieval coverage rates and improvements in the classification of relevant documents on these two systems. These results are very important in real-world scenarios where testers don't always have time to analyze all existing documents.

Keywords: thesaurus creation; information retrieval; query expansion; text mining; software testing.

LISTA DE FIGURAS

Figura 1 –	Etapas do processo de MT ……..……..	16
Figura 2 –	Processo de tokenização de uma frase ……...	19
Figura 3 –	Processo de POS-tagging …….	20
Figura 4 –	Processo de eliminação de stopwords ……...	21
Figura 5 –	Processo de Stemização ……	23
Figura 6 –	Arquitetura de SRI: Fase 1 - Criação do Índice ……..	26
Figura 7 –	Estrutura de índices Invertidos: Fase 1 - Criação do Índice …….	28
Figura 8 –	Cálculo da tf normalizada: Fase 1 - Criação do Índice …….	28
Figura 9 –	Arquitetura de SRI: Fase 2 - Recuperação de Informação ……..	29
Figura 10 –	Etapas do Processo de clusterização ……..	34
Figura 11 –	Estrutura de um sistema de Extração de Informação baseado em PLN ……...	36
Figura 12 –	Etapas para geração automática de tesauro usando métodos estatísticos …….	45
Figura 13 –	Etapas para geração automática de tesauro linguístico …….	46
Figura 14 –	Relacionamento entre conceitos no WordNet ……..	48
Figura 15 –	Diagrama de fluxo de dados do processo …….	53
Figura 16 –	Trecho do Mapa Mental ……...	55
Figura 17	Trecho do grafo RDF do tesauro de domínio de Conhecimento/Negócios ……..	65
Figura 18	Modelo da estrutura do Thesa ……	66
Figura 19 –	Diagrama com visão geral de uso do protótipo ……	68
Figura 20 –	Interface principal do sistema …….	69
Figura 21 –	Ambiente de criação e atualização do tesauro ……...	70

LISTA DE TABELAS

Tabela 1 – Desempenho comparativo do sistema AVS …….. 73

Tabela 2 – Valores médios de métricas em relação às 20 CRs melhor ranqueadas ……..... 74

Tabela 3 – Desempenho comparativo do ATP com 10 RNs …….. 76

Tabela 4 – Desempenho comparativo do ATP com 4 RNs ……..... 77

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizagem de Máquina
CR	Change Request
CT	Caso de Teste
EI	Extração de Informação
IA	Inteligência Artificial
MT	Mineração de Texto
PLN	Processamento de Linguagem Natural
RI	Recuperação de Informação
RN	Release Notes
SKOS	Simple Knowledge Organization System
SMT	Sistema de Mineração de Texto
SRI	Sistema de Recuperação de Informação
SW	Software

SUMÁRIO

1 INTRODUÇÃO 12
1.1 TRABALHO REALIZADO 13
1.2 ESTRUTURA DO DOCUMENTO 14

2 MINERAÇÃO DE TEXTO 15
2.1 ETAPAS DA MINERAÇÃO DE TEXTO 16
2.2 COLETA 17
2.3 PRÉ-PROCESSAMENTO 17
2.3.1 Tokenização 18
2.3.2 Part-of-speech tagging 19
2.3.3 Remoção de stopwords 20
2.3.4 Redução ao radical/Stemming 22
2.3.5 Distância de edição 23
2.4 INDEXAÇÃO E RECUPERAÇÃO DE INFORMAÇÃO 25
2.4.1 Fase 1 – Indexação da Base de Documentos 26
2.4.2 Fase 2 – Consulta e Recuperação de Documentos 29
2.4.3 Modelos de RI 30
2.4.4 Ferramenta Solr 30
2.5 MINERAÇÃO – EXTRAÇÃO DE PADRÕES 31
2.5.1 Classificação e Agrupamento 32
2.5.1.1 Classificação 32
2.5.1.2 Agrupamento 34
2.5.2 Extração de Informação 35
2.6 ANÁLISE **36**
2.7 CONSIDERAÇÕES FINAIS 38

3 CONSTRUÇÃO E USO DE TESAUROS 40
3.1 DEFINIÇÕES GERAIS 40
3.2 CONCEITOS BÁSICOS 41
3.3 TIPOS DE TESAUROS E MÉTODOS DE CONSTRUÇÃO 42
3.3.1 Tesauros Estatísticos 43
3.3.1.1 Análise local versus Análise global 43
3.3.1.2 Métodos Estatísticos 44
3.3.2 Tesauros Linguísticos 46
3.3.2.1 Métodos de Construção de Tesauros Linguísticos 46
3.3.2.2 WordNet 47
3.4 USO DE TESAUROS EM ENGENHARIA DE SOFTWARE 48

4 UM PROCESSO PARA CONSTRUÇÃO DE TESAUROS DE DOMÍNIO ESPECÍFICO 50
4.1 TESAUROS PARA AUXILIAR PROCESSOS DE TESTE DE SW 50
4.2 VISÃO GERAL DO PROCESSO 52
4.3 BASES DE DADOS UTILIZADAS NO PROCESSO 54
4.3.1 Mapa Mental 54
4.4 ETAPAS DE PROCESSAMENTO 56
4.4.1 Criação do corpus de documentos 56
4.4.2 Módulo de Pré-processamento 57
4.4.3 Criação do tesauro com palavras isoladas (Processo I) 58
4.4.4 Derivação de Termos Compostos (Processo II) 59
4.4.4.1 Criação de Bigramas 60
4.4.4.2 Criação de Concatenações, Abreviações e Acrônimos 60
4.4.5 Unificação dos synsets criados 61
4.4.6 Validação dos synsets criados 62
4.4.7 Indexação do Tesauro criado 63
4.5 TRABALHOS RELACIONADOS 64
4.5.1 Um método para criação de tesauros semânticos de requisitos usando a ontologia SKOS 64
4.5.2 THESA: ferramenta para construção de tesauro semântico aplicado interoperável 65
4.6 CONSIDERAÇÕES FINAIS 66

5 MOTOSAURUS - ESTUDO DE CASO 67
5.1 MOTOSAURUS – UM TESAURO PARA O DOMÍNIO MOTOROLA 67
5.1.1 Interface Principal do Motosaurus 69
5.1.2 Interface de Criação, Atualização e Validação 70
5.2 TESTES COMPARATIVOS 71
5.2.1 O Sistema AVS 71
5.2.1.1 Testes comparativos 72
5.2.2 O Sistema ATP 74
5.2.2.1 Testes comparativos 75
5.3 CONSIDERAÇÕES FINAIS 78

6 CONCLUSÃO 79
6.1 PRINCIPAIS CONTRIBUIÇÕES 79
6.2 Trabalhos Futuros 80

REFERÊNCIAS 82

1 INTRODUÇÃO

Grandes empresas de desenvolvimento de software costumam manter repositórios com milhares de documentos textuais (e.g., documentos de requisitos, casos de teste, solicitações de mudança de código, entre outros). Contudo, em geral, essas empresas não adotam regras ou diretrizes sobre como nomear termos específicos de domínio da aplicação. Assim, devido à falta de um vocabulário padrão, muitas vezes esses documentos utilizam palavras distintas para se referir à mesma entidade no domínio da aplicação. Embora os funcionários conheçam os termos padrão, o uso de abreviações, acrônimos e concatenações arbitrárias para se referir à mesma entidade é algo comum dentro de empresas de SW.

Essas variações de terminologias dificultam o processamento automático dos textos, que se mostra uma atividade central nos contextos onde documentos são gerados e processados a cada dia. Esse problema se torna ainda mais visível no cenário atual, no qual as empresas vêm utilizando sistemas de Recuperação de Informação (RI) para facilitar a localização rápida de documentos relevantes para uma diversidade de tarefas. Tais sistemas são baseados em palavras-chaves, tanto na indexação como na recuperação de documentos. Contudo, termos usados nas consultas nem sempre estão presentes nos documentos relevantes (exemplo, "foto" x "image"). Assim, a consulta não irá recuperar todos os documentos relevantes existentes, a fim de garantir a finalização adequada da atividade em curso.

Segundo Bascher & Café (2010), uma solução frequentemente adotada para melhorar o desempenho de sistemas de processamento de texto é a utilização de dicionários de sinônimos (Tesauros). Esses dicionários consistem em conjuntos de termos equivalentes (termos sinônimos ou relacionados). Tesauros são frequentemente usados para melhorar o desempenho de sistemas de RI textuais, geralmente através da expansão automática das consultas construídas pelo usuário (Baeza-Yate & Ribeiro-Neto, 2013). Essa técnica adiciona termos correlacionados à consulta original, ampliando as chances do sistema em recuperar documentos relevantes. Vale ressaltar que os tesauros têm outros usos importantes no processamento de texto, como por exemplo melhorar o desempenho de classificadores de texto ou prover um vocabulário padrão, entre outros.

Embora existam diferentes dicionários de sinônimos disponíveis, eles são geralmente genéricos - por exemplo, WordNet (Miller, 1995). Assim, eles não são apropriados para aplicações de domínio específico, principalmente devido à ambiguidade lexical (uma palavra com duas ou mais interpretações semânticas diferentes). Para domínios particulares, faz-se necessário um dicionário mais refinado, com termos que ocorrem apenas dentro do domínio da aplicação. Nesse contexto, este trabalho propõe um processo para a construção de tesauros de domínio específico, com o objetivo de evitar o uso de termos ambíguos ou não correlacionados no domínio, encontrados em tesauros genéricos (como o WordNet).

1.1 TRABALHO REALIZADO

Este documento apresenta um processo para a geração de tesauros de domínio específico, criados a partir de documentos disponíveis nos repositórios das empresas. O processo recebe como entrada documentos textuais selecionados, e extrai desses documentos os termos relevantes para o domínio da Empresa. Em seguida, os termos relacionados são agregados em conjuntos.

O processo utiliza técnicas de Processamento de Linguagem Natural (PLN) (Juraska et al., 2018) como POS-tagging, RegEx e n-gramas para enriquecer o dicionário de sinônimos com termos compostos (e.g., "Inteligência Artificial"), siglas (e.g., "IA") e abreviações. Finalmente, os conjuntos de saída de palavras/termos relacionados são indexados e disponibilizados para consulta manual ou automática.

Este trabalho foi desenvolvido no contexto de um projeto de cooperação de pesquisa em Teste de SW entre a Motorola Mobility (uma empresa Lenovo) e o CIn-UFPE. Como tal, nosso estudo de caso se concentrou no domínio de teste de SW.

O protótipo implementado recebe como entrada documentos textuais, como Solicitações de Mudança e descrições de Casos de Teste, a partir dos quais o dicionário de sinônimos é derivado. O tesauro construído como resultado deste trabalho foi usado na expansão automática de consultas para melhorar o desempenho de dois sistemas de RI da Motorola Mobility.

Testes comparativos revelaram que o tesauro melhorou a cobertura de recuperação e a classificação de documentos relevantes nesses dois sistemas utilizados na empresa. Esses resultados são muito importantes em cenários do mundo real, nos quais os testadores nem sempre têm tempo para analisar todos os documentos existentes.

Este trabalho deu origem a uma publicação no Simpósio Brasileiro de Engenharia de Software, ocorrido em setembro de 2021 (Carvalho, Barros & Prudêncio, 2021).

1.2 ESTRUTURA DO DOCUMENTO

O presente trabalho está dividido em mais 5 capítulos, além da Introdução, brevemente descritos a seguir.

O Capítulo 2 apresenta uma breve revisão sobre a Mineração de Texto, destacando suas fases. O texto dá ênfase às etapas de mineração mais intimamente relacionadas ao nosso trabalho.

O Capítulo 3 discute a construção e o uso de tesauros, apresentando métodos de construção baseados nas abordagens estatística e linguística. Também apresentamos brevemente o uso de tesauro na Engenharia de Software.

O Capítulo 4 apresenta o processo proposto de construção de tesauros para domínio específico. Inicialmente, apresentamos o contexto de desenvolvimento do trabalho e como os tesauros podem auxiliar no processo de teste de software. A seguir, damos uma visão geral do processo proposto, seguida de um detalhamento das bases de dados usadas, bem como das etapas de processamento.

O Capítulo 5 apresenta as funcionalidades do protótipo e o estudo de caso realizado utilizando duas ferramentas da Empresa parceira.

O Capítulo 6 conclui o documento, apresentando as principais contribuições e sugestões de trabalhos futuros.

2 MINERAÇÃO DE TEXTO

Mineração de textos (MT), também conhecida como descoberta de conhecimento em textos é um processo que visa extrair informações importantes e desconhecidas em documentos textuais (Pezzine, 2016). Esse processo utiliza técnicas de Processamento de Linguagem Natural e métodos analíticos para encontrar e extrair padrões e tendências em textos não estruturados (Kadhim, 2019). Diferentemente da Mineração de Dados, que trabalha com dados estruturados, como por exemplo tabelas de bancos de dados, a MT é voltada para textos sem estruturação rígida, que geralmente são escritos em linguagem natural.

O surgimento da MT foi impulsionado pela necessidade de descobrir padrões e irregularidade em textos de maneira automatizada, sendo vista como uma extensão da área de mineração de dados, porém, com foco na análise textual (Aranha & Passos, 2006). Segundo Allahyari et al. (2017), os processos de MT permitem classificar, agrupar e extrair informação do texto, bem como criar taxonomias (ontologias), sumarizar textos e minerar opiniões (análise de sentimento). Para Aranha & Passos (2006), o crescimento no armazenamento de dados digitais não estruturados, que não são diretamente processados por computadores para extração de conhecimento, propiciou o uso das técnicas de mineração de texto.

Considerada um campo multidisciplinar, a MT envolve áreas da informática, como Processamento de Linguagem Natural (PLN) e Estatística (Gupta e Lehal, 2009). Através de uma dessas áreas, ou da convergência entre elas, é possível transformar um texto em linguagem natural em uma estrutura que uma máquina possa compreender (Hotho; Nurnberger & Paass, 2005). Segundo Aranha & Passos (2006), diferentemente de um mecanismo de busca (e.g., Google[1]), onde o usuário sabe o que deseja encontrar, a MT busca informações desconhecidas para o usuário.

[1] Google: www.google.com

2.1 ETAPAS DA MINERAÇÃO DE TEXTO

Segundo Corrêa et al. (2012), o processo de MT pode conter várias etapas, que podem ser divididas em: limpeza dos documentos, seleção, integração, transformação, mineração, avaliação dos modelos encontrados e apresentação do conhecimento adquirido. Já para Aranha & Passos (2006), o processo de MT é subdividido em cinco etapas, que devem ser executadas de forma encadeada. Para Martins et al. (2003), apenas quatro etapas são necessárias no processo de MT, sendo elas: coleta de documentos, pré-processamento, extração de conhecimento e avaliação e interpretação dos resultados.

Apesar de alguns autores citarem a indexação como etapa no processo de MT, é possível realizar diversas atividades sem que precise indexar os dados. Segundo Chatterjee et al. (2020) uma etapa que é imprescindível no processo de MT é o pré-processamento, uma vez que, os dados de textos devem estar limpos para os algoritmos possam obter uma melhor análise do documento textual. A Figura 1 mostra as etapas principais da MT, levando em consideração a abordagem definida por Aranha (2007).

Figura 1 – Etapas do processo de MT

Fonte: autor (2021)

Como é possível observar, todos os autores descrevem o processo de MT como etapas interativas que vão transformando os dados de entrada até se obter o objetivo desejado. Na metodologia de Silva et al., (2016), é interessante a utilização

de um corpus textual para extrair conhecimento dentro de um domínio particular, que possui grandes volumes de textos escritos em linguagem natural.

Para este trabalho, utilizamos a abordagem bastante conhecida descrita em Aranha (2007) como base, ressaltando apenas uma reestruturação simples do processo apresentado na figura 1 (ver Capítulo 4).

2.2 COLETA

A coleta de dados é a etapa responsável pela formação do conjunto de documentos que irão compor o *corpus* (i.e., coleção de documentos). Os documentos selecionados para compor o *corpus* devem estar diretamente relacionados ao domínio do conhecimento a ser extraído, para não originar dados irrelevantes e inconsistentes no processo de Mineração (Pezzini, 2017).

Segundo Aranha (2007), a base que compõe o corpus pode ser adquirida de forma manual, realizada por um ser humano, ou automática por meio de um crawler[2] , podendo ser uma base estática, nos casos mais simples, ou dinâmica, isto é, atualizada frequentemente. O processo de criação desse corpus pode ser demorado, uma vez que, na sua grande maioria, exige atividades manuais e necessita de um especialista do domínio de interesse (Indurkhya & Damerau, 2010).

2.3 PRÉ-PROCESSAMENTO

Segundo Jung & Lee (2020), os textos utilizados pelos sistemas de MT devem sofrer alguma preparação antes de serem enviados para os algoritmos de descoberta de conhecimento. É necessário realizar alguns procedimentos (pré-processamento) para preparar os dados textuais, a fim de facilitar a interpretação automática do texto.

O pré-processamento tem por objetivo criar representações internas bem estruturadas, de modo a alimentar os algoritmos subsequentes (Jung & Lee, 2020). A preparação dos documentos de entrada é necessária para que os próximos

[2] **Crawler** - programa de computador que navega pela Web de uma forma metódica e automatizada, coletando dados/arquivos de interesse da aplicação que controla o *crawler*.

processos sejam capazes de manipular todos os documentos da maneira correta (Pezzini, 2017).

Nesse contexto, o Processamento de Linguagem Natural (PLN) é uma área de grande importância para a MT, tendo como propósito fornecer aos computadores a capacidade de reconhecer, interpretar e/ou gerar textos em alguma língua natural de forma automática (Kaur & Madan, 2015). Segundo Correia (2018), faz-se necessário utilizar algumas técnicas de Processamento de Linguagem Natural (PLN) na etapa de pré-processamento para realizar a normalização do conteúdo dos documentos. Essas técnicas auxiliam na padronização da representação dos textos, permitindo uma manipulação adequada e contribuindo para o reconhecimento claro de seu significado.

De acordo com Medeiros (2010), as técnicas de PLN provêm diversos benefícios no processo de descoberta do conhecimento. Com o uso de processadores morfossintáticos (parsers), é possível agregar ao texto informações como a classe gramatical das palavras (do inglês, Part-Of-Speech – seção 2.3.4), ou mesmo a função de cada palavra na frase (sujeito, objeto etc). Também é possível reduzir as palavras ao seu radical (stemm – seção 2.3.3) para simplificar a representação dos textos. Além dessas, existem ainda recursos/técnicas para descrever informações semânticas presentes nos termos que podem ser usadas em corretores gramaticais, mineração de opiniões, tradução automática de texto, entre outros (Liu & Zhang, 2013).

Embora a fase de pré-processamento não siga regras fixas, existem alguns processos que são comumente utilizados nessa etapa (Alexandersson & Nyholm, 2007). As seções a seguir apresentam com mais detalhes os principais processos desta etapa da MT: tokenização, remoção de stopwords, stemming e Part-of-speech tagging. A criação da representação vetorial do documento será vista na seção 2.4.

2.3.1 Tokenização

A primeira fase do pré-processamento é a identificação/delimitação das frases e palavras (Tokenização). Em alguns casos, são removidos sinais de pontuação e caracteres especiais, como também é eliminada a diferenciação entre letras maiúsculas e minúsculas.

Segundo Allahyari et al. (2017), a tokenização é o processo de transformar o texto em um conjunto de tokens. Cada token pode ser representado por uma palavra (que são os termos simples) ou por um agrupamento de tokens (termos compostos). O objetivo desse procedimento é deixar o texto mais limpo/conciso (Medeiros, 2010).

Com a identificação dos tokens, é possível representar os textos como dados estruturados (vetores de termos) em dimensões possíveis de analisar (Jackson & Moulinier, 2007). A Figura 2 exemplifica o processo de tokenização.

Figura 2 – Processo de tokenização de uma frase

Frase exemplo:

The camera is not working when disabled permission to the picture

Tokenização de termos:

|The||camera| |is| |not| |working| |when| |disabled| |permission| |to||the| |picture|

Fonte: Autor (2021)

Em geral, os tokens são identificados e isolados a partir de delimitadores como espaço em branco e sinais de pontuação. Porém, é importante notar que algumas línguas não utilizam o espaço como delimitador, como por exemplo o Japonês e o Chinês (Damerau, 2010).

Ao final desse processo, cada documento de entrada passa a ser representado como uma lista de termos no estilo Bag of Words (isto é, ainda podendo conter palavras/termos duplicados). Essa lista de termos poderá ainda ser submetida a vários outros processos, como os apresentados nas seções a seguir.

2.3.2 Part-of-speech tagging

Segundo Collobert et al. (2011), Part-of-speech (PoS) tagging é o procedimento de rotular palavras com marcação de referência, indicando sua classe gramatical - por exemplo, substantivo, verbo, advérbio, etc. A marcação da classe gramatical das palavras é uma tarefa importante em muitas aplicações, tais como corretores gramaticais, extração de informação, análise de sentimento/mineração de opinião, tradução automática, entre outras (Fonseca et al. 2015).

Figura 3 – Processo de POS-tagging

Frase exemplo:
The camera is not working when disabled permission to the picture
Representação pós-POS-tagging:

[The, DT], [camera, NN], [is, VBZ], [not, RB], [working, VBG], [when, WRB], [disabled, JJ], [permission, NN], [to, TO], [the, DT], [picture, NN]

Fonte: Autor (2021)

A etiquetagem apresentada na figura 5 foi realizada com o uso da ferramenta PoS-tagging de livre uso[3]. O significado de cada etiqueta pode ser encontrado no site da ferramenta. Segundo Collobert et al. (2011), os PoS-taggers podem ser construídos usando diferentes abordagens. Alguns são baseados em regras gramaticais, outros utilizam modelos de aprendizado de máquina, sendo esta segunda opção a mais utilizada.

Muitos algoritmos têm sido propostos e aplicados com sucesso nesta tarefa, alcançando bons resultados em diferentes línguas. Ainda assim, a marcação das classes gramaticais continua sendo explorada, em busca de algoritmos que melhorem a precisão dessa marcação automática. Para Jones & Somers (2013), a ambiguidade natural nos idiomas muitas vezes leva os algoritmos a rotularem algumas palavras de maneira errada. Note que, na realidade, PoS-taggers são classificadores multiclasse (uma vez que a mesma palavra pode ser classificada em mais de uma classe gramatical – e.g., lancha (barco - substantivo) e lancha (verbo flexionado - lanchar)).

2.3.3 Remoção de stopwords

Segundo Baeza-Yate & Ribeiro-Neto (2013), as *stopwords* são termos que aparecem com grande frequência em documentos, e que não acrescentam valor semântico à análise textual, pois são palavras com pouca relevância para a recuperação de documentos (Brito, 2016). É comum encontrar em listas de *stopwords* artigos, preposições, conjunções e pronomes, visto que essas palavras aparecem com alta frequência nos textos, porém não apresentam grande relevância

[3] https://www.nltk.org/book/ch05.html

semântica. O procedimento de remoção de palavras irrelevantes reduz o tamanho da representação (dimensionalidade) dos textos (lista de termos), e consequentemente reduz a estrutura de indexação dos documentos (Baeza-Yate & Ribeiro-Neto, 2013) – seção 2.4. A remoção de stopwords também melhora a acurácia de classificadores de texto baseados em Aprendizagem de Máquina.

Apesar de existirem stoplists (listas de stopwords) padrão disponíveis em diferentes idiomas, elas podem ser definidas/modificadas de acordo com a aplicação. Geralmente, um especialista no domínio define os termos que serão utilizados na stoplist da sua aplicação, para só então a remoção ser realizada de forma automática. Segundo Zaman, Matsakis & Brown (2011), é necessário ter cuidado ao gerar uma stoplist, pois caso sejam removidas palavras que tenham importância para a interpretação do texto, o resultado do processamento dos dados pode ser prejudicado.

Na Figura 4 é apresentado um exemplo do processo de remoção de stopwords após a tokenização. Note que, quando a eliminação de stopwords é feita apenas com base na stoplist, a etiquetagem das classes gramaticais (PoS-tagging) não interfere nesse processo. Assim, a fim de simplificar o exemplo, não incluímos a informação de classe gramatical na representação no quadro da figura 4. Contudo, também é possível eliminar stopwords com base nessa informação (por exemplo, eliminar todos os artigos e preposições). Isso é uma decisão do administrador do sistema. Por fim, vale salientar que, independente do processo de eliminação de stopwords adotado, essa fase deve ser realizada após a etapa de PoS-tagging, uma vez que essa etiquetagem necessita da frase completa para ter bons resultados.

Figura 4 – Processo de eliminação de stopwords

Frase exemplo:
The camera is not working when disabled permission to the picture
Representação pós-tokenização:
|The||camera| |is| |not| |working| |when| |disabled| |permission| |to||the| |picture|

Remoção de stopwords:
|The||camera| |is| |not| |working| |when| |disabled| |permission| |to||the| |picture|

Representação pós-stopwords:
|camera| |working||disabled| |permission||picture|

Fonte: Autor

De acordo com a Figura 4, é possível observar que, após a remoção das stopwords, houve uma redução da lista de palavras que representa a frase original. Assim, esse procedimento reduz a dimensionalidade da representação dos textos (listas de termos oriundos da tokenização).

2.3.4 Redução ao radical/Stemming

O processo de stemming tem por objetivo reduzir uma palavra ao seu radical (stem), reduzindo variações morfológicas à sua forma raiz. Assim, é possível unificar em um só token as palavras derivadas do mesmo radical (ligadas, assim, ao mesmo conceito) (Soares, 2010). Por exemplo, as palavras “prato”, “pratinho” e “pratão” seriam representadas pelo token “prat”. Como exemplo na língua inglesa, os termos “works”, “working” e “worked” serão reduzidos a sua forma raiz “work” após o processo de stemming.

Assim como a eliminação de stopwords, essa técnica também reduz a dimensionalidade da representação dos documentos. Segundo Correia (2018), com a utilização desta técnica, a lista de tokens é reduzida, favorecendo a eficiência na etapa de Indexação de documentos (seção 2.4).

Os algoritmos usados neste processo não levam em consideração o contexto no qual a palavra ocorre, nem sua classe gramatical (PoS). Em geral, os algoritmos determinam a raiz de uma palavra com a eliminação de prefixos e sufixos[4], como plural e flexões verbais (De Ávila & Soares, 2013). A figura 3 exemplifica o processo de stemming com o mesmo exemplo citado nas seções anteriores.

É importante salientar que o stem da palavra nem sempre é idêntico ao seu radical morfológico (raiz). A raiz morfológica das palavras é obtida através da aplicação de regras morfológicas de redução[5], que são mais sofisticadas do que apenas a eliminação de sufixos. Porém, na prática os sistemas de PLN utilizam a operação de stemming por ser bem mais simples de implementar e por oferecer resultados de eficácia equivalentes.

---

[4] Stemming - https://en.wikipedia.org/wiki/Stemming

[5] https://brasilescola.uol.com.br/gramatica/estrutura-e-formacao-de-palavras-i.htm

Segundo Gomes (2009), cada idioma possui suas próprias regras de stemming. Segundo Ali & Ibrahim (2012), existem diversos algoritmos de stemming, sendo o Porter[6] o mais conhecido. O Porter é um algoritmo simples, bastante utilizado na língua inglesa, baseado em regras que determinam quais sufixos devem ser removidos (Willet, 2006).

Figura 5 – Processo de stemização

Frase exemplo:

The camera is not working when disabled permission to the picture

Representação pós-tokenização:
|The||camera| |is| |not| |working| |when| |disabled| |permission| |to||the| |picture|

Representação pós-stopwords:
|camera| |working||disabled| |permission||picture|

Stemização dos termos:
|camera| |working||disabled| |permission||picture|

Representação pós-stemização:
|camera| |work||disabl| |permiss||pictur|

Fonte: Autor (2021)

Apesar de a operação de stemming ser bastante útil na maioria dos casos, é importante destacar que, dependendo do domínio e da aplicação final, a precisão dos resultados obtidos pode ser prejudicada (Sohrawardi; Azam; Hosain, 2014). Isso se deve ao fato de que alguns termos serão aglutinados em um mesmo token, perdendo suas características distintivas. É recomendado analisar os efeitos causados por este processo, para não perder eficácia e eficiência em sistemas de indexação e busca, bem como de classificação e agrupamento (clustering). Essa questão ficará mais clara na seção 2.4, a seguir.

2.3.5 Distância de edição

Dado um conjunto de strings, uma função (ou métrica) de Distância calcula o menor custo possível associado às operações de edição necessárias para

[6] https://tartarus.org/martin/PorterStemmer/

transformar uma dada cadeia de caracteres em outra (Lisbach & Meyer, 2013). Quanto menor o valor da distância, maior a similaridade entre as duas palavras.

As operações de edição usadas por algumas funções de distância são compostas por um conjunto de quatro regras, sendo elas:

- Inserção de caracteres ausentes na string;
- Exclusão de um caractere presente na string;
- Substituição de um caractere na string por outro caractere;
- Transposição ou troca de posição de dois caracteres presente na string.

As operações descritas acima não necessariamente precisam ser todas usadas para verificar a similaridade entre strings. De fato, as operações podem ser combinadas e o valor obtido pela função é dado pela soma dos custos individuais das operações utilizadas.

Alguns exemplos de métricas que são utilizadas para medir similaridade entre strings são apresentadas a seguir.

Distância de Hamming (Sankoff & Kruskal, 1983) é usada para medir a distância entre duas Strings[7] de comprimento igual. Essa métrica utiliza apenas operações de substituição, e atribui um custo fixo igual a 1 para cada diferença encontrada. A função é definida da seguinte maneira:

$$dist_{Ham}(X_{1...c}, Y_{1...s}) = \begin{cases} 0 & \text{se } c = 0 \text{ e } s = 0 \\ dist_{Ham}(X_{1...c-1}, Y_{1...s-1}) + & \\ \beta(x_c, y_s) & \text{se } c = s \text{ e } c, s > 0 \\ \infty & \text{se } c \neq s \end{cases}$$

Distância de Levenshtein (Levenshtein, 1966), também conhecida como cálculo de edição, pode analisar a distância entre strings com comprimentos iguais ou diferentes. Essa métrica permite calcular o número mínimo de inserções, substituições e exclusões necessárias para transformar uma cadeia de caracteres em outra. A função é definida da seguinte maneira:

[7] String: uma sequência de caracteres, geralmente utilizada para representar palavras.

$$dist_{Lev}(X_{1...c}, Y_{1...s}) = \begin{cases} \max\{c, s\} & \text{se } c = 0 \\ & \text{ou } s = 0 \\ \min \begin{cases} dist_{Lev}(X_{1...c-1}, Y_{1...s-1}) + \beta(x_c, y_s) \\ dist_{Lev}(X_{1...c}, Y_{1...s-1}) + 1 \\ dist_{Lev}(X_{1...c-1}, Y_{1...s}) + 1 \end{cases} & \text{se } c, s > 0 \end{cases}$$

A Distância de Levenshtein tem como principal aplicação determinar o quão similar duas strings podem ser, como feito em verificadores de ortografia e gramática. Essa métrica, ao contrário da distância de Hamming, pode medir a distância entre duas strings independente de seu tamanho, sendo bastante eficiente para comparação de strings pequenas.

Coeficiente de Dice (Dice, 1945) é uma métrica baseada em caracteres. Os caracteres de uma string é comparado com o de outra string e os valores podem variar entre zero a 1, sendo 1 quando as sequências de entrada são exatamente iguais. A medida de similaridade entre duas strings é definida como duas vezes o número de caracteres comuns das strings dividido pelo número total de caracteres testados em ambas entidades. A função é definida conforme a equação abaixo:

$$sim_{Dice}(X, Y) = \frac{2 * Q(X \cap Y)}{Q(X) + Q(Y)}$$

Um fator positivo em utilizar métricas sobre strings é devido ao cálculo ser direto, ou seja, as cadeias de caracteres são comparadas de forma direta, sem necessitar de representações intermediárias da coleção de documentos, como o modelo do espaço vetorial. Em coleções de documentos que sofrem alterações, a representação vetorial precisa ser novamente gerada a cada mudança que ocorre na coleção, diferentemente das medidas de similaridade entre string, que não sofrem com este problema.

2.4 INDEXAÇÃO E RECUPERAÇÃO DE INFORMAÇÃO

A Recuperação de Informação (RI) é uma área da computação que lida com armazenamento, organização e acesso a documentos, provendo aos usuários a recuperação automática de informação de seu interesse (Cardoso, 2004). Segundo

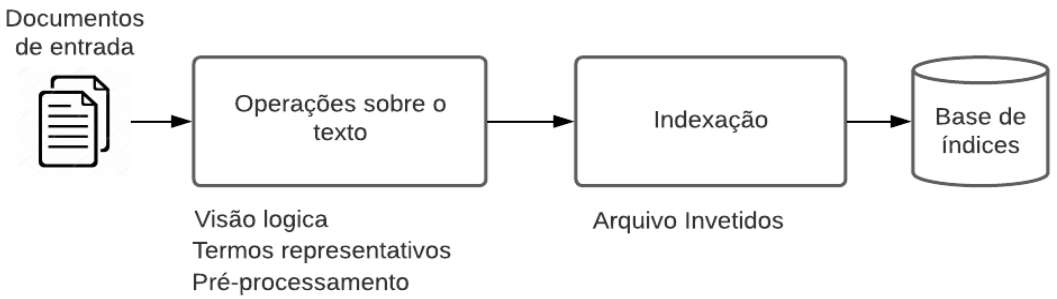
González (2003), os sistemas de RI são utilizados como um ambiente intermediário entre o solicitante e a fonte de informação. Para Frantz (1997), um sistema de RI tem como principal objetivo atender a necessidade de informação do usuário. Esses sistemas geralmente é denominados de “engenhos de busca”, podendo ser de acesso restrito (e.g., um sistema de empresa), ou uso livre, como o sistema Google[8].

Os sistemas de RI são compostos por diversos componentes, tendo duas fases principais (Baeza-Yates e Ribeiro-Neto, 2011): a indexação dos documentos (Fase 1) e a consulta à base de documentos indexados (Fase 2). A primeira fase lida com a preparação e a indexação da coleção de documentos, resultando em uma base de documentos indexados por palavras-chave (a base de índices invertidos). A segunda fase lida com a recuperação de informação, oferecendo acesso aos documentos indexados através de uma interface de consultas. Essa segunda fase inclui algoritmos que processam as consultas e fazem a ordenação dos documentos recuperados de acordo com sua relevância em relação aos termos da consulta. Os resultados são apresentados ao usuário através da interface de consultas.

2.4.1 Fase 1 – Indexação da Base de Documentos

De acordo com Baeza-Yates & Ribeiro-Neto (2013), a indexação é um mecanismo que converte o conteúdo dos documentos em um formato que facilita a busca e recuperação das informações nos documentos de maneira rápida e precisa. A figura 6 apresenta o ciclo básico da Fase 1 de um sistema de RI.

Figura 6 - Arquitetura de SRI: Fase 1 - Criação do Índice



Fonte: autor (2021)

[8] https://www.google.com/

Esse processo gera a Base de Índices, também chamada de “arquivo invertido”, na qual cada documento é indexado pelas palavras/termos que ele contém para posterior consulta pelos usuários (Correia, 2018).

Como vemos na Figura 6, o passo inicial na criação de um sistema de RI é a seleção dos documentos que serão indexados e armazenados na base (o corpus de documentos). Como já foi descrito na seção 2.2, a coleta dos documentos pode ser manual ou automática, realizada por um crawler.

Segundo Correia (2018), faz-se necessário um pré-processamento da estrutura textual dos documentos que foram previamente selecionados para que o processo de indexação funcione de forma precisa. O pré-processamento pode ser realizado de forma manual, quando os termos são manualmente tratados, ou automaticamente por um sistema de RI.

Os sistemas de RI podem indexar o texto completo (do inglês, full-text indexing), ou podem executar “Operações sobre o texto” para realizar seleção automática de termos, seguindo as etapas de pré-processamento desejadas, já apresentadas na seção 2.3 (Blumetti et al., 2007). Essas operações reduzem a lista de tokens que representam cada documento (também chamado de vetor de termos), melhorando o desempenho dos sistemas de RI.

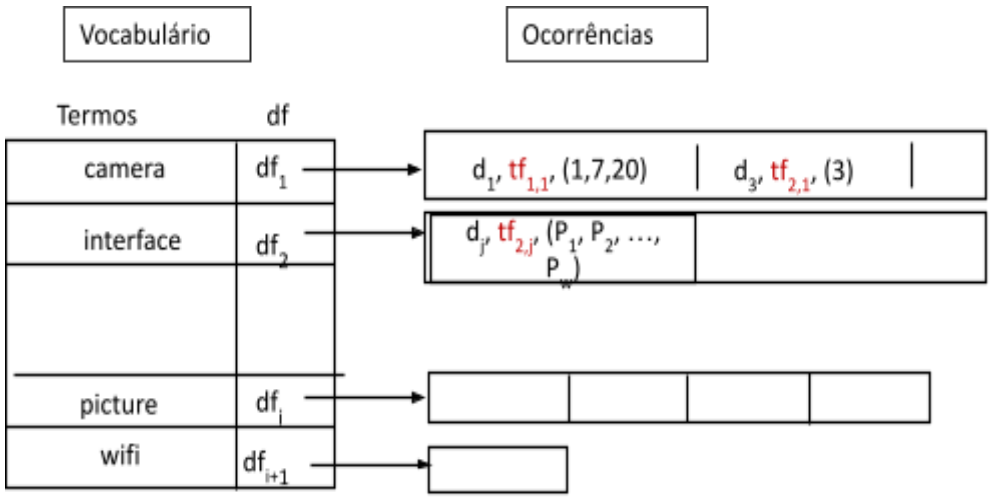
A seguir, o sistema de RI cria o dicionário de termos da base (Vocabulário da base), que consiste em uma lista com todas as palavras-chave diferentes que ocorrem na base de documentos sendo indexada. O Vocabulário da Base é obtido através da união das listas de tokens que representam os documentos de entrada. Assim, esse vocabulário contém todos os termos relevantes (oriundos do pré-processamento) que ocorrem nos documentos da Base.

Por fim, vem a indexação dos documentos, através da criação dos arquivos invertidos. A estrutura dos arquivos invertidos é composta por dois elementos principais: o vocabulário da base e os arquivos de endereçamento, que guardam as ocorrências de cada termo nos documentos da base (Baeza-Yate & Ribeiro-Neto, 2013).

Cada entrada da estrutura do Vocabulário guarda informações sobre a palavra-chave (como por exemplo, quantos documentos na base contêm essa

palavra – do inglês, *document frequency - df*), e um link para uma entrada correspondente no arquivo de endereçamentos, como detalhado a seguir (Figura 7).

Figura 7 – Estrutura de índices Invertidos: Fase 1 - Criação do Índice



Fonte: autor (2021)

Geralmente, os arquivos de endereçamento são formados por diversas listas com identificadores (endereços) dos documentos que contêm o termo corrente (suas ocorrências). Além do endereço do documento em questão, cada ocorrência (célula) na lista pode armazenar também a frequência do termo naquele documento (*term frequency - tf*) e a posição no documento de cada ocorrência desse termo. No exemplo da figura 7, o termo “câmera” ocorre 3 vezes em d_1 nas posições (1,7,20).

Geralmente, o *tf* é representado pela frequência normalizada do termo no documento (ver figura 7), onde $freq_{i,j}$ é a frequência do termo k_i no documento d_j e $max_l\, freq_{l,j}$ é a frequência do termo mais frequente no documento d_j.

Figura 8 – Cálculo da *tf* normalizada: Fase 1 - Criação do Índice

$$tf_{i,j} = \frac{freq_{i,j}}{max_l\, freq_{l,j}}$$

Fonte: autor (2021)

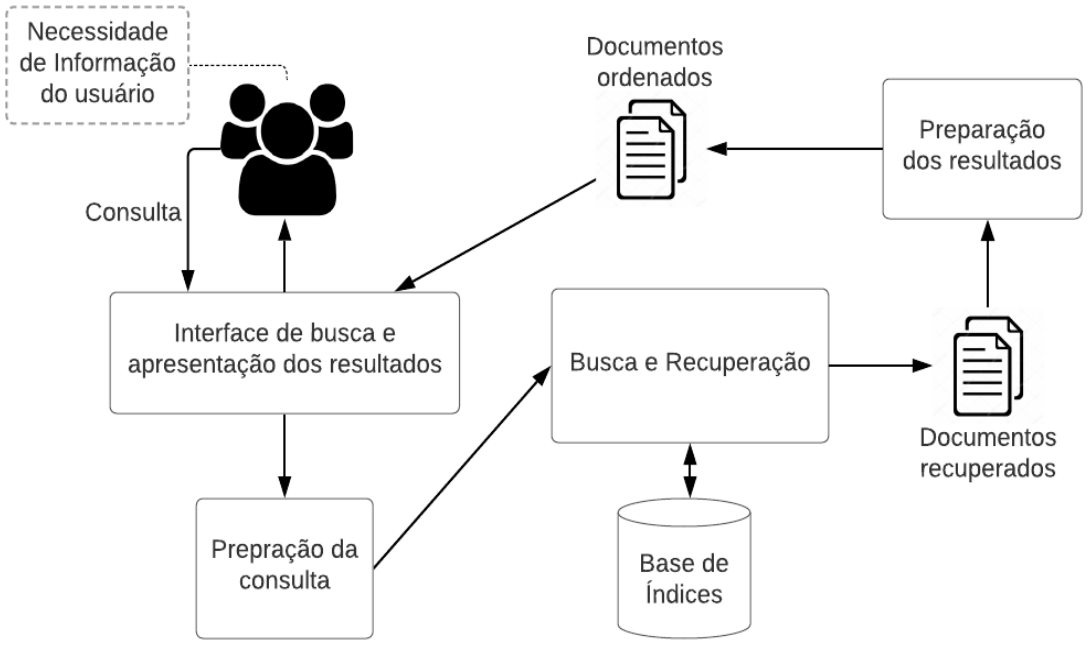
Essas informações serão úteis na Fase 2 dos sistemas de RI (seção 2.4.2), facilitando a recuperação dos documentos relevantes, pois só é necessário acessar a lista de ocorrências para identificar que documentos contêm uma determinada

palavra (Gonzalez, 2003; Baeza-Yate & Ribeiro-Neto, 2013). Já a informação sobre a posição das palavras no documento é utilizada para responder consultas que buscam por sequências de termos consecutivos (consultas com contexto, geralmente entre aspas).

2.4.2 Fase 2 – Consulta e Recuperação de Documentos

Na fase de recuperação, o usuário digita suas consultas, e o sistema de RI retorna os documentos relevantes (que contêm as palavras da consulta do usuário). A figura 7 apresenta o ciclo básico da fase 2 de um sistema de RI.

Figura 9 - Arquitetura de SRI: Fase 2 - Recuperação de Informação



Fonte: autor (2021)

Em geral, os documentos selecionados pelo sistema são ordenados de acordo com sua relevância em relação à consulta. Uma das maneiras mais simples de medir relevância é verificar a quantidade de vezes que cada palavra da consulta ocorre nos documentos (*df*, já apresentado na seção 2.4.1). Quanto maior sua frequência, maior a relevância do termo para representar o documento. Contudo, nem todos os sistemas de RI oferecem essa possibilidade, como veremos a seguir.

2.4.3 Modelos de RI

Cada sistema de RI adota um modelo interno para representação/indexação dos dados. Destacam-se aqui três modelos clássicos que são comumente utilizados nos sistemas de RI (Baeza-Yate & Ribeiro-Neto, 2013): o modelo Booleano, o modelo espaço Vetorial e o modelo Probabilista. Cada modelo tem suas características, que impactam no funcionamento do sistema como um todo.

O modelo Booleano permite a utilização de operadores booleanos na consulta (Soares, 2009). Porém, esse modelo não armazena a frequência de ocorrência dos termos nos documentos (*tf*). Em consequência disso, ele não é capaz de ordenar os documentos de acordo com sua relevância.

O modelo Espaço Vetorial, por sua vez, armazena as frequências dos termos e usa isso como pesos para ordenar a lista de documentos retornados (Leskovec; Rajaraman; Ullman, 2014). Esse modelo é o mais difundido e utilizado em sistemas de RI, por ser simples e muito eficaz.

No modelo Probabilista, a classificação dos documentos é feita em ordem decrescente de relevância probabilística, utilizando cálculos mais complexos baseados na teoria da probabilidade (Bonfim, 2009).

2.4.4 Ferramenta Solr

Dentro da Mineração de Textos, os sistemas de RI são de grande valor para facilitar o acesso rápido e preciso a documentos/informações que serão posteriormente analisados por outros processos de mineração, como veremos nas próximas seções. A importância da RI e sua eficácia impulsionaram o desenvolvimento de diversos buscadores de documentos com diferentes modelos. Várias soluções comerciais e open-source[9] foram desenvolvidas com o intuito de solucionar ou minimizar os problemas de RI. Entre elas, podemos destacar o Solr, um software open-source mantido pela Fundação Apache que possui a capacidade de indexar documentos e fazer buscas de forma rápida e eficiente.

[9] Open source: código-fonte de um software, que pode ser adaptado para diferentes fins.

O Apache Solr[10] é uma plataforma de pesquisa corporativa popular, extremamente rápida e de código aberto, desenvolvida no Apache Lucene (Shahi, 2015). Foi desenvolvido como um servidor independente que possui uma interface gráfica de administrador baseado na web, permitindo realizar pesquisas comuns e tarefas relacionadas a indexação, consulta, análise de linguagem e muitas outras.

Segundo Grainger & Potter (2014), o Solr permite realizar pesquisas em textos estruturados e não estruturados. Possui uma interface de exibição de resultados das buscas no formato facetado (categorização do conjunto de resposta de uma consulta). Também é possível fazer integração com banco de dados, buscas geoespaciais e análise do conteúdo dos documentos (XLS, DOC, PDF e MP3). A ferramenta permite fazer configurações em arquivos para definir os metadados, ou seja, como o texto deve ser tokenizado durante a indexação e a consulta. Também existe a possibilidade de usar a ferramenta sem definição dos metadados, e permitir que o próprio Solr crie campos enquanto indexa os dados.

O Solr foi desenvolvido utilizando a tecnologia Java, e atua como um servidor de pesquisa de texto autônomo, sendo executado dentro de um contêiner servlet[11]. Por meio do Apache Lucene, o Solr indexa textos completos e realiza pesquisas sem precisar implementar técnicas de busca com algoritmos complexos. Possui APIs que podem ser integradas com outras linguagens de programação (exemplo: Python[12], Java, NET, etc), e se adapta a qualquer tipo de aplicação, mesmo sem codificação Java. Sua arquitetura permite personalizar suas funcionalidades com plugins ou extensões, deixando a aplicação mais dinâmica.

2.5 MINERAÇÃO – EXTRAÇÃO DE PADRÕES

A etapa de Mineração é a mais importante de todo o processo de MT, podendo também ser chamada de Extração de Padrões. Segundo Fayyad et al. (1996), com o processo de Mineração, é possível identificar padrões válidos existentes nos dados. Um padrão é considerado “válido” quando os casos relacionados ao padrão encontrado são consistentes, trazendo informações novas,

[10] Solr: https://lucene.apache.org/solr/
[11] servlet: classe Java usada para estender as funcionalidades de um servidor.
[12] Python: Linguagem de programação. Site https://www.python.org/.

úteis e compreensíveis. Além disso, para que os padrões sejam de fato úteis, eles devem ser descritos usando alguma representação que possa ser interpretada pelos usuários do sistema de Mineração.

As tarefas comuns da etapa de Mineração são: classificação e agrupamento, extração de informação e extração de regras de associação. Contudo, nem todo sistema de Mineração de Texto executa todas essas tarefas obrigatoriamente, sendo possível selecionar a técnica/tarefa de mineração que melhor se adeque às necessidades do usuário (Brito, 2016).

As seções a seguir apresentam brevemente cada uma dessas tarefas, trazendo também algumas informações sobre as técnicas usadas em cada tarefa.

2.5.1 Classificação e Agrupamento

Para muitas aplicações de MT, faz-se necessário realizar classificação e/ou agrupamento dos textos de entrada. A escolha de qual tarefa será realizada depende do problema em questão. Se o objetivo é avaliar o grau de semelhança entre dados textuais sem conhecimento prévio das suas classes, a tarefa de agrupamento (*clustering*) é a mais recomendada. Caso o objetivo seja categorizar um conjunto de documentos de acordo com classes já conhecidas (pré-definidas), a tarefa indicada é a classificação. Para Brito (2016), a natureza do problema vai definir quais as tarefas que devem ser realizadas para atingir o objetivo desejado.

2.5.1.1 Classificação

A classificação é um procedimento no qual itens individuais são organizados em grupos de classes pré-estabelecidas com base em informações quantitativas sobre uma ou mais características específicas aos itens (Russo, 2008). Sendo também uma das áreas de interesse da Recuperação de Informação, a classificação textual identifica a classe de um documento a partir do seu conteúdo, provendo um meio de organizar as informações de forma eficiente e inteligente, a fim de facilitar sua posterior recuperação (Madureira, 2009).

Segundo Sebastiani (2002), duas abordagens distintas de Inteligência Artificial (IA) têm sido adotadas para desenvolver classificadores de texto: Engenharia do conhecimento e Aprendizagem de máquina. Ambas serão descritas

abaixo. Note que, independente da abordagem adotada, a preparação dos documentos através de um pré-processamento é importante para melhorar a precisão dos classificadores.

Engenharia de conhecimento (EC) - dentro dessa abordagem, um especialista no domínio define manualmente regras que serão usadas para classificar os documentos a partir de características presentes no texto. Dependendo da implementação do sistema, os metadados (campos descritivos) contidos nos documentos podem ser considerados. Para Cazes e Feitosa (2018), o processo de classificar com regras explícitas deve ser semelhante à classificação manual, e o conjunto de regras definidas pelo especialista deve se assemelhar ao raciocínio humano usado durante o processo manual.

A estrutura de classificação baseada em regras é composta por uma condicional “SE condição ENTÃO conclusão”, assemelhando-se a uma regra de associação (Camilo, 2009). Em geral, as regras utilizam os termos mais frequentes em cada classe para determinar a categoria de cada documento.

O trabalho de identificação dos termos característicos de cada classe pode ser feito de modo manual, por um especialista, ou com auxílio de ferramentas da área de EC. Além disso, em alguns casos as regras também podem ser derivadas utilizando-se algoritmos da área de Aprendizagem de Máquina (MCCUE, 2007 apud CAMILO; SILVA, 2009). Destacamos aqui as árvores de decisão, os algoritmos de Cobertura Sequencial (*Sequential Covering Algorithm*).

Aprendizagem de Máquina (AM) - nesta abordagem, o sistema é capaz de adquirir conhecimento de forma automática. Para isso, os algoritmos buscam regularidades nos dados e tomam decisões baseadas em experiências acumuladas através da solução bem sucedida de problemas anteriores (Monard & Baranauskas, 2003). A Aprendizagem de Máquina pode ser realizada com duas abordagens principais: aprendizado supervisionado, quando se dispõe de um conjunto de dados etiquetados; e aprendizado não supervisionado, quando não há etiquetagem do conjunto de dados.

A aprendizagem supervisionada baseia-se em conjuntos de dados etiquetados, nos quais cada instância possui uma variável-alvo pré-definida,

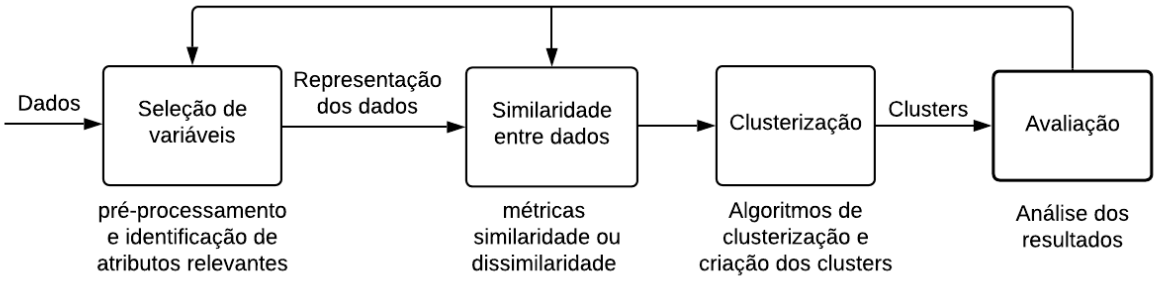
chamada de *target*. Os algoritmos de AM realizam procedimentos para identificar as características dos conjuntos de dados que foram etiquetados. A partir daí, esses algoritmos realizam indução de regras e cálculos probabilísticos, para então atribuir o *target* (classe) correspondente a uma nova instância.

Na aprendizagem não supervisionada, por outro lado, não existe a indução de regras. Os algoritmos não possuem informações sobre a classe à qual pertence cada instância. Essa abordagem busca encontrar similaridades ou diferenças entre os documentos do conjunto de entrada, a fim de identificar atributos úteis à sua classificação ou detectar tendências.

2.5.1.2 Agrupamento

A tarefa de agrupamento (em inglês, clustering) tem o objetivo de dividir o conjunto inicial de dados em grupos distintos, de modo que cada grupo contenha documentos similares entre si e diferentes dos documentos nos outros grupos (Xu e Wunsch, 2008). Diferente do processo de classificação, o agrupamento de dados é uma subcategoria da aprendizagem não supervisionada que busca estabelecer uma regularidade estatística entre os dados de entrada, formando representações internas que codifiquem as características dos dados e criem novas classes ou grupos automaticamente (Rojas, 1996). A figura 8 mostra de forma genérica o procedimento de agrupamento.

Figura 10: Etapas do Processo de clusterização



Fonte: Adaptado de Jain et al. (1999)

Os algoritmos de agrupamento utilizam alguns critérios para verificar a similaridade entre os dados, critérios esses capazes de medir o quanto dois

documentos são semelhantes entre si, e se devem ou não ser adicionados ao mesmo grupo (Lopes, 2004). Entre os métodos utilizados para comparar dados, temos os que utilizam medidas de distância aplicáveis a espaços vetoriais, como por exemplo: distância Euclidiana, distância de Manhattan e similaridade do Cosseno.

2.5.2 Extração de Informação

A Extração de Informação (EI) é uma área que se propõe a extrair palavras-chave ou padrões de informação de maneira automática a partir de documentos textuais estruturados, semiestruturados ou não estruturados (Appelt & Israel, 1999). Segundo Cowie & Lehnert (1996), os sistemas de EI são construídos para compreender textos e extrair dados relevantes que atendam à necessidade de informação de um usuário.

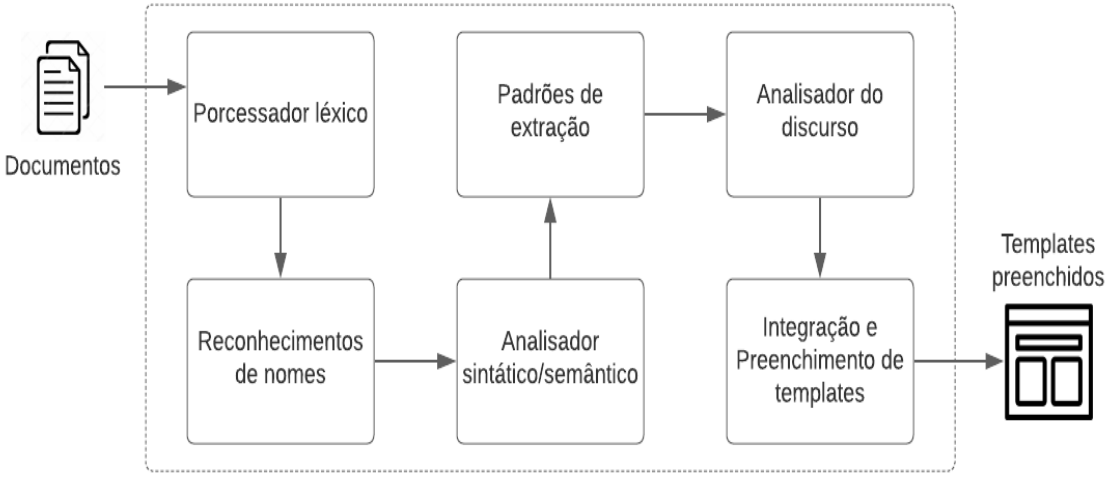
A informação a ser extraída é determinada por um conjunto de regras e padrões de extração, que geralmente são definidos manualmente por um especialista do domínio, ou com diferentes graus de automação utilizando técnicas da Aprendizagem de Máquina. Com a AM, a indução das regras de extração é automatizada e a participação humana é minimizada (Glickman & Jones, 1999).

Segundo Silva (2004), os sistemas de EI podem ser divididos em duas abordagens principais: *Wrappers* e sistemas baseados em PLN.

Wrappers (programas extratores) são desenvolvidos para processar textos estruturados e semiestruturados. Esses sistemas se baseiam na formatação do texto, em tags, expressões regulares, frequência de palavras etc, sendo muito utilizados para tratar documentos Web. *Wrappers* podem ser construídos manualmente por um especialista no domínio, ou podem ser criados a partir da utilização de algoritmos de AM. Geralmente, essa escolha vai depender dos dados disponíveis para treinar os algoritmos de AM.

A segunda abordagem mais utilizada na EI é baseada em técnicas de Processamento da Linguagem Natural, sendo capazes de tratar textos livres (sem estrutura definida). Esses sistemas executam um pré-processamento linguístico no texto, para então realizar a extração dos dados relevantes. A arquitetura de um sistema de EI baseado em PLN está representada na Figura 7.

Figura 11: Estrutura de um sistema de Extração de Informação baseado em PLN



Fonte: Adaptado de Grishman (1997).

Embora a AM seja largamente utilizada (Krishnan & Manning, 2006), existem outras técnicas que são amplamente adotadas em sistemas de EI, como por exemplo as expressões regulares (RegEx), criadas com o auxílio da ferramenta[13] Python. Segundo Simões et al. (2018), RegEx (Expressões regulares) são extratores de informação textuais que podem ser utilizados nos sistemas de EI baseados em regras, podem reconhecer instâncias de entidades simples que podem prover a identificação de relacionamentos entre entidades mais complexas.

2.6 ANÁLISE

Nesta etapa, são examinados os resultados do processo de mineração de texto. O conhecimento extraído na etapa de mineração é visualizado e interpretado, para avaliar se o resultado do processo resultou na descoberta de um novo conhecimento a partir dos dados coletados (Brito, 2016). Esses procedimentos são necessários para compreender e interpretar o conhecimento descoberto para, em seguida, avaliar o quão útil ele será (Passini, 2012).

A avaliação dos classificadores é feita através da matriz de confusão. A partir dessa matriz, é possível identificar a acurácia da classificação dos itens categorizados. Veja um exemplo de uma Matriz de confusão binária para M = 2, onde M é o número de rótulos possíveis.

[13] https://www.nltk.org/book/ch03.html

Previsto - Real	Positivo	Negativo
Positivo	VP	FP
Negativo	FN	VN

- **Verdadeiro positivo (VP):** quando a classe positiva foi prevista corretamente;
- **Falso positivo (FP):** quando a classe positiva foi prevista incorretamente;
- **Falso verdadeiro (FV):** quando a classe negativa foi prevista corretamente;
- **Falso negativo (FN):** quando a classe negativa foi prevista incorretamente.

Outras métricas importantes derivadas da matriz de confusão são: Acurácia, Precisão, Recall e F1-score.

Acurácia - refere-se à quantidade de itens corretamente classificados como positivos ou negativos. Em geral, medidas de acurácia são aquelas cujo valor se aproxima do valor correto, ou seja, valor de referência. Porém, podem ocorrer diversas limitações como, por exemplo, a possibilidade de enviesar o entendimento, quando o conjunto de dados não é balanceado.

$$\text{Acurácia} = \frac{VP + VN}{VP + FP + VN + FN} = \frac{\text{Total de Acertos}}{\text{Total de dados do conjunto}}$$

Precisão - refere-se à quantidade de dados corretamente classificados como positivos com relação ao conjunto total de dados classificados como sendo positivos. Assim como a Acurácia, a Precisão também é suscetível ao desbalanceamento do conjunto de dados.

$$\text{Precisão} = \frac{VP}{VP + FP} = \frac{\text{Total de positivos classificado corretamente}}{\text{Total de predições positivas}}$$

Recall - refere-se à taxa de dados classificados como positivos comparada com o total de positivos no corpus.

$$\text{Recall} = \frac{VP}{VP + FN} = \frac{\text{Total de positivos classificado corretamente}}{\text{Total de positivos}}$$

F1-Score - refere-se ao cálculo da média harmônica entre Precisão e Recall, retornando um valor único que indica a qualidade geral do classificador.

$$\text{F1-score} = 2 * \frac{\text{Precisção} * \text{Recall}}{\text{Precisção} + \text{Recall}}$$

O cálculo de F1-score visa entender o classificador sobre os dados positivos. Quando seu score é máximo (1/1), o classificador é considerado perfeito.

Além do que foi visto acima, existem outras medidas que podem ser utilizadas para avaliar classificadores. Por exemplo, a Análise ROC (ou curva ROC), do inglês *Receiver Operating Characteristic.* Trata-se de um método gráfico que avalia, organiza e seleciona sistemas estatísticos de predição e diagnóstico (Egan, 1975). Destaca-se aqui também o Índice Kappa, usado para descrever e testar o grau de concordância entre a classe predita e a observada na classificação (Matos, 2009), entre outros. No entanto, nosso foco foi nas medidas de precisão e cobertura, pois ambas atendem satisfatoriamente a avaliação do trabalho proposto.

2.7 CONSIDERAÇÕES FINAIS

Neste capítulo, foram apresentados os conceitos básicos sobre a Mineração de Texto, tornando claro que essa área oferece uma abordagem eficiente para obter conhecimentos ocultos em documentos textuais. Destacamos que o pré-processamento é um procedimento que contribui de forma significativa para uniformização do conteúdo de documentos livres, tornando os dados padronizados para serem submetidos aos algoritmos de mineração.

A combinação de técnicas usadas para extrair informações pode contribuir para melhorar a qualidade do resultado final. As ideias apresentadas neste capítulo foram estruturadas de modo a definir os procedimentos para o desenvolvimento de

um protótipo que incorpore os conhecimentos acima citados, e de maneira simples e direta resolva as necessidades impostas pelos usuários.

3 CONSTRUÇÃO E USO DE TESAUROS

Os tesauros são dicionários de sinônimos (ou palavras relacionadas), podendo ser gerais de uma língua ou específicos para um dado domínio. Esses dicionários são usados tanto na indexação de documentos, quanto na recuperação mais eficaz da informação. Sua definição, processos de construção e usos serão detalhados neste capítulo.

3.1 DEFINIÇÕES GERAIS

A palavra *tesauro* (“thesaurus”) é de origem latina e significa “tesouro”, tendo sido utilizada no sentido de “tesouro de palavras” (Pereira, 2005). O termo se popularizou em 1852 com a publicação da obra de Peter Mark Roget "Thesaurus of English Words and Phrases", com sua primeira versão editada (Foskett, 1985). Essa publicação objetivou organizar as palavras de forma a facilitar a expressão de ideias.

Em 1950, o pesquisador da IBM Hans Peter Luhn percebeu que, para encontrar a palavra/ideia certa para recuperar informações, não bastava apenas ter uma lista de palavras em ordem alfabética. Seria necessário utilizar uma estrutura em que as palavras tivessem referências cruzadas, a fim de mostrar quais ideias estavam interligadas. Isso levou à criação de listas controladas de palavras, chamadas de tesauros (Dodebei, 2002; Granada, 2011).

Segundo Baeza-Yates & Ribeiro-Neto (2013), o tesauro envolve uma normalização do vocabulário, e inclui uma estrutura mais complexa, que vai além de uma simples lista de palavras e seus sinônimos. De maneira geral, o tesauro não só inclui palavras isoladas, termo simples, mas também termos compostos, formados normalmente por grupos nominais que representam um determinado conceito em um determinado domínio do conhecimento (por exemplo, Inteligência Artificial – área de pesquisa e desenvolvimento). No contexto de indexação e recuperação de informação, os principais objetivos de um tesauro são: (a) prover um vocabulário padrão, (b) ajudar usuários na localização de termos para criar consultas, e (c) prover relações hierárquicas e associativas para atender as necessidades de informação de um usuário.

A função do tesauro é ajudar nas buscas realizadas em sistema de RI e representar o conteúdo dos documentos (Gomes, 2009). Ainda segundo a autora, os tesauros podem ser monolíngues (apenas um idioma) ou multilíngues (quando são criados em uma língua, mas possuem versões em outros idiomas). Além disso, existem tesauros multidisciplinares, que compreendem conceitos de diferentes domínios, ou tesauro aplicados a um contexto específico. Segundo Duarte (2007), o tesauro é um instrumento que, quando construído, possibilita o controle terminológico e a padronização das linguagens utilizadas por usuários e por indexadores de documentos.

Segundo Café (2010), a padronização facilita o processo de indexação e recuperação da informação, controlando a ambiguidade das palavras, como sinonímia (um conceito representado por duas ou mais palavras, por exemplo, cão e cachorro) e homonímia (uma palavra que expressa mais de um conceito, por exemplo, tênis – calçado- e tênis –esporte). Os relacionamentos entre os termos são um aspecto importante do tesauro que contribui de maneira significativa para recuperação de documentos relevantes.

3.2 CONCEITOS BÁSICOS

Um Tesauro pode ser definido como uma lista pré-compilada de termos organizados em conjuntos cujos elementos possuem relacionamentos de similaridade semântica entre si, podendo assim representar um único “conceito”. O termo escolhido para representar o conjunto (i.e., o conceito) é chamado “descritor”, enquanto os termos relacionados são denominados “não descritores”.

Os tipos de relacionamentos que podem caracterizar um tesauro, segundo Miranda (1990), são: relação hierárquica, de equivalência e associativa.

Relação Hierárquica: ocorre entre dois termos diferentes com características gerais idênticas, sendo que um deles representa o conceito mais amplo, deixando-o em um nível superior ao outro (e.g., felino – gato). Segundo Campos & Gomes (2006), a relação hierárquica pode ser considerada quando se observa relação entre dois termos que possuem finalidade de posição em uma estrutura sistemática; ou

seja, quando existe precedência entre dois conceitos, podendo um deles incluir o escopo total do outro.

Relação de Equivalência: ocorre quando dois ou mais termos representam o mesmo conceito. Quando dois termos equivalentes estão presentes no mesmo tesauro, um deles é designado como termo preferido. Veja alguns exemplos de sinônimo que resultam em relações de equivalência:

- Nomes populares e científicos (e.g., *abacaxi* e *Ananas comosus*);
- Nomes comuns e científicos e marcas comerciais (e.g., Gillette e Lâmina de barbear);
- Abreviaturas e nomes complexos (e.g., IA e Inteligência Artificial) .

Relação Associativa: identifica ligação entre conceitos que não possuem relação de equivalência ou hierárquica, porém mantêm algum tipo de vínculo espacial ou temporal, seja de causa/efeito, produto, ou etapas de um processo. Segundo Dahlberg (1978), a relação associativa ocorre entre termos do tipo funcional, que não possuem relacionamentos determinados pelas características contidas nos conceitos (por exemplo, causa/efeito => acidente/lesão). Geralmente, a relação é estabelecida para identificar dependências entre conceitos, e os critérios de associação entre termos devem ser explícitos, claros e precisos, para evitar definições superficiais.

Usar categorias para enquadramento de conceitos permite organizar os termos e refinar o posicionamento de termos associados a cada conceito. Segundo Van der Laan (2002), o relacionamento semântico entre conceitos é estabelecido e dividido entre as relações hierárquicas e associativas. Já as relações de equivalência estabelecem relação entre termos que descrevem os mesmos conceitos. O construtor do tesauro deve também observar os casos de palavras que possuem relação de equivalência com termos ambíguos, podendo então incluí-las ou retirá-las do tesauro.

3.3 TIPOS DE TESAUROS E MÉTODOS DE CONSTRUÇÃO

Baeza-Yates & Ribeiro-Neto (2013) destacam dois tipos de tesauros no contexto de sistemas de RI: estatísticos (seção 3.3.1) e linguísticos (seção 3.3.2).

Tesauros linguísticos estão mais baseados em relações sintáticas entre termos, considerando o seu uso na língua escrita e falada. Já os tesauros estatísticos criam associações entre termos correlacionados no contexto de um determinado corpus.

Os tesauros podem ser gerais ou específicos para um domínio de aplicação, podendo ser criados de forma manual ou automática. Essa escolha deve levar em consideração os aspectos metodológicos e operacionais do processo de construção, de acordo com a linha adotada para sua criação. Neste trabalho, iremos apresentar apenas os métodos automatizados, uma vez que essa foi a nossa abordagem escolhida. Vale ressaltar que o processo de criação manual de tesauros apresenta uma demanda maior de conhecimento especialista humano (linguístico), ocasionando um aumento de tempo e esforço na sua construção.

3.3.1 Tesauros Estatísticos

Esses tesauros são construídos com base na coocorrência entre termos dentro de um corpus de documentos. Um termo pode ser uma letra, um sigla (e.g., IA), uma palavra ou um termo composto (e.g., Inteligência Artificial), sendo chamado de Vocabulário o conjunto de todos os termos que ocorrem no corpus. Nessa abordagem, uma matriz de correlação é construída a partir dos documentos. A matriz computa quantas vezes cada termo ocorre junto com outro no corpus total, podendo realizar uma análise global (que considera todos os documentos do corpus ao mesmo tempo), ou análise parcial/local (que considera o contexto atual, por exemplo, o texto processado até o momento).

Cada grupo que compõe o tesauro é construído a partir dos valores obtidos na matriz de correlação. Se duas palavras ocorrem frequentemente juntas, o sistema cria relação entre elas. Por exemplo, os termos “Inteligência Artificial” podem ocorrer frequentemente juntos em um corpus de documentos sobre essa área de pesquisa. Como esses termos são fortemente relacionados, quando o usuário digitar a palavra “Inteligência” na consulta, o sistema irá expandi-la com a palavra “Artificial”.

3.3.1.1 Análise local *versus* Análise global

A construção de tesauros estatísticos é baseada em etapas que identificam e extraem os termos do corpus através de processos que variam de acordo com a

abordagem adotada. Em alguns casos, é utilizada a abordagem de Análise Automática Global, que considera todos os termos contidos no corpus ao mesmo tempo. Porém, existem métodos que adotam a abordagem de Análise Automática Local, na qual é utilizado um subconjunto do corpus de documentos, produzindo tesauros mais específicos.

No contexto de sistemas de RI, a análise local constrói uma matriz de correlação no momento em que a consulta é processada (Baeza-Yates & Ribeiro-Neto, 2013). Inicialmente, o usuário envia uma consulta ao sistema de RI que, por meio de um processo automático, seleciona os “n” primeiros documentos recuperados que obtiveram melhores classificações. Esses documentos constituem o corpus a partir do qual o tesauro temporário será construído. Portanto, trata-se de um conjunto "local" de documentos específicos para uma determinada consulta. A ideia é obter uma maior precisão no processo de recuperação, uma vez que esses métodos consideram apenas documentos relevantes no contexto da consulta atual.

Na análise global, a associação entre termos considera a coocorrência dos termos em todos os documentos do corpus. Os métodos apresentados a seguir realizam análise automática global do corpus para gerar o tesauro.

3.3.1.2 Métodos Estatísticos

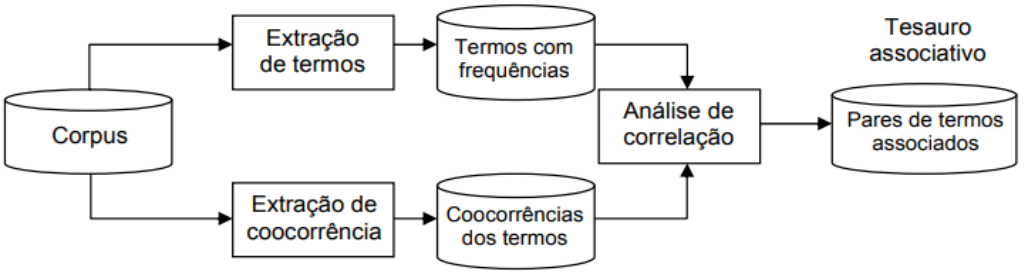
Como já mencionado, os tesauros estatísticos são gerados a partir de coocorrencia entre termos. A matriz de correlação quantifica o grau de associação entre termos que aparecem juntos. Segundo Park, Han & Choi (1996), existem diversas medidas de similaridade na literatura que podem ser utilizadas para quantificar associações entre termos. A medição de similaridade é uma característica comum em abordagens estatísticas, sendo geralmente usada para identificar termos que podem estar associados dentro de um corpus.

Em Kaji et al. (2000), a geração automática de um tesauro estatístico é feita através de uma abordagem conhecida como associação de primeira ordem. Esta técnica propõe que a similaridade semântica possa ser computada pela relação lexical entre vizinhos. Por exemplo, as cores “vermelho” e “azul” ocorrem frequentemente junto com palavras como “cor”, “flor”, “carro”, “escuro”, e assim por

diante. Desse modo, “vermelho” e “azul” serão consideradas palavras semanticamente relacionadas.

O modelo de tesauro proposto por kaji et al. (2000) possui algumas etapas, como: extração de termos, extração de coocorrência dos termos e análise de correlação (ver figura 8).

Figura 12: Etapas para geração automática de tesauro usando métodos estatísticos



Fonte: Adaptado de kaji et al. (2000).

O autor afirma que os termos que melhor representam os conceitos são os substantivos simples ou compostos que ocorrem com frequência no corpus. Para selecionar os termos compostos é utilizado um método de desambiguação estrutural no qual termos composto mais frequentes são comparados com termos compostos menos frequentes. A regra geral para a desambiguação estrutural de termos compostos é: “se existe um termo composto CN que inclui dois termos compostos CN1 e CN2, sendo esses termos incompatíveis entre si, então é selecionado o termo com mais frequência como candidato a ser escolhido”.

No cálculo de coocorrência, são extraídos pares de termos que aparecem juntos dentro de uma “janela” que se move ao longo do texto, e é constituída por um conjunto de cadeias de *n* palavras, onde *n* é um número previamente escolhido. Esses pares de palavras ainda passam por uma etapa de filtragem para eliminar redundâncias. O tamanho da janela pode ser definido arbitrariamente, considerando também o custo computacional. Os autores definiram janelas com 20 a 50 palavras.

A análise de correlação entre termos feita pelos autores é baseada na medida de Informação Mútua, descrita no trabalho de Church & Hanks (1990). Essa medida pré determina o grau de similaridade por um número máximo de termos associados a cada termo no corpus. Isso permite escolher os termos que irão compor o tesauro.

3.3.2 Tesauros Linguísticos

Esta abordagem utiliza o contexto sintático para calcular a similaridade entre termos. Contexto sintático pode ser entendido como as relações entre os termos de um segmento de texto. Para estabelecer esse contexto, é utilizado um analisador sintático para identificar as categorias gramaticais dos termos. Com os termos previamente etiquetados, é então possível identificar as relações sintáticas entre termos, como sujeito, objeto direto ou indireto de um verbo, etc.

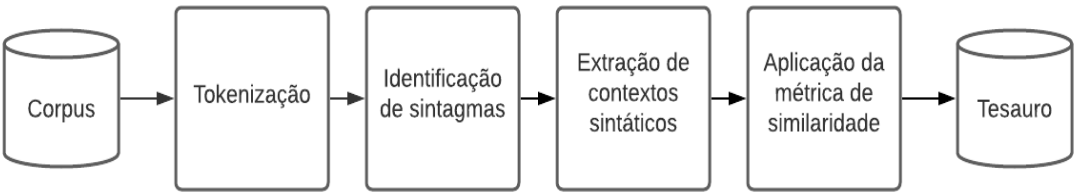
Os tesauros linguísticos também podem ser gerais ou específicos para um determinado domínio. Os tesauros de domínio específico contém apenas termos relacionados ao domínio escolhido (vocabulário controlado). Como exemplo de tesauro geral, pode-se citar o WordNet. Essa base lexical contém quase todos os termos da língua inglesa, e para cada entrada, o WordNet fornece sinônimos (termos relacionados), hipônimos (termos especializados) e hiperônimos para expandir, especificar e generalizar a consulta, respectivamente.

A seguir, discutiremos métodos para construção de tesauros linguísticos. Já a seção 3.3.2.2 apresenta com mais detalhes o tesauro WordNet, uma vez que ele é utilizado por nosso processo de geração de tesauro específico.

3.3.2.1 Métodos de Construção de Tesauros Linguísticos

Como já mencionado, a criação dos tesauros linguísticos utiliza informações sintáticas para obter similaridade entre termos. Um dos primeiros trabalhos a utilizar informações sintáticas para a construção de um tesauro foi proposto por Grefenstette (1994). O autor descreve que é possível criar tesauro a partir de um corpus usando informações sintáticas presentes nos termos. Para criar o tesauro o autor realiza uma série de passos que podem ser vistos na figura 9.

Figura 13: Etapas para geração automática de tesauro linguístico



Fonte: Adaptado de Grefenstette (1994).

O processo proposto por Grefenstette (1994) é iniciado com a tokenização dos termos. Em seguida, é aplicado um analisador sintático (e.g., PoS-tagging) para rotular as classes morfossintáticas. O analisador faz a identificação de sintagmas nominais e verbais para obtenção do contexto sintático de cada termo. Os sintagmas permitem identificar os contextos sintáticos dos termos, por exemplo, se um substantivo está ligado a um verbo como sujeito ou como objeto.

Por fim, é utilizada uma medida de similaridade que calcula o peso global e local de cada termo em seu contexto sintático. Para calcular o peso global, é verificada a quantidade de termos do corpus associados ao contexto. Para o peso local, leva-se em consideração a frequência do contexto sintático que modifica um determinado termo. Depois de estimar os pesos globais e locais, é realizada uma multiplicação desses pesos, para então utilizar uma variante da medida de Jaccard (Tanimoto, 1958 apud Grefenstette, 1994) entre os termos e gerar a lista de termos semelhantes que irão compor o tesauro.

Embora existam diferentes abordagens de construção de tesauros linguísticos, a grande maioria utiliza como referência o trabalho de Grefenstette (1994). A extração de informações sintáticas dos termos e a utilização de métricas de similaridade para agrupamento de termos são fatores cruciais para construir tesauros dessa natureza.

3.3.2.2 WordNet

WordNet[14] é um banco de dados lexical para a língua inglesa desenvolvido na Universidade de Princeton. Os termos são distribuídos em quatro classes gramaticais: substantivos, verbos, adjetivos e advérbios.

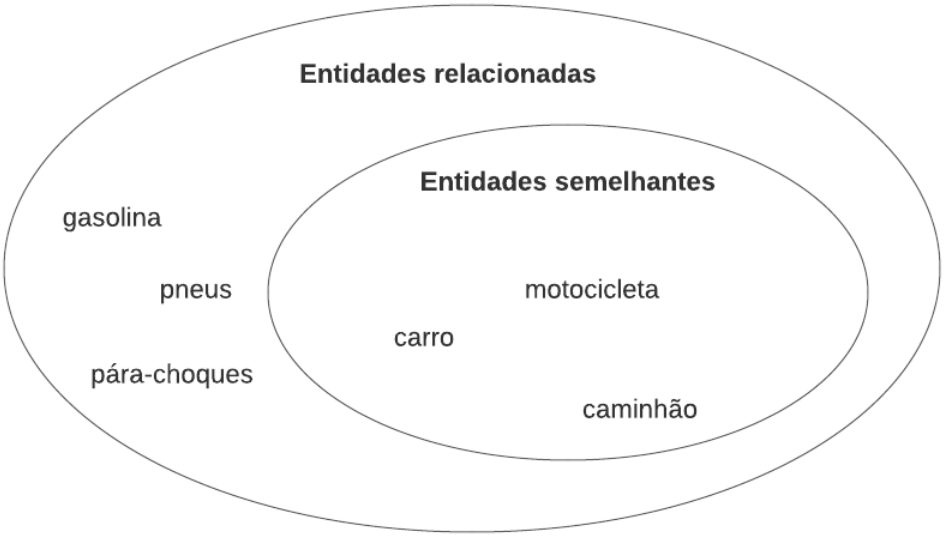
Sua plataforma pode ser acessada por meio de um navegador da Web, assemelhando-se a um tesauro eletrônico, ou seja, um dicionário eletrônico de sinônimos e antônimo com relações de natureza lógica e conceitual. Sua estrutura é considerada uma ontologia lexical na qual os conceitos, chamados de “synsets” (conjuntos de sinônimos), se conectam por meio de tipos específicos de relacionamento. Os synsets (forma abreviada do termo inglês synonym sets) representam os conceitos das unidades lexicais da língua, sendo interconectados

[14] https://wordnet.princeton.edu/

por relações semântico-conceituais e lexicais que definem domínio conceitual com suas particularidades, termos e relações relevantes. Suas relações semânticas são hipernímia/hiponímia (é um), meronímia (é parte de), sinonímia e antonímia.

O conjunto de synsets é materializado em estruturas de grafos onde os vértices representam os synsets e as arestas representam as relações entre eles (Miller, 1995). Cada synset está relacionado a outros synsets maiores ou menores, que correspondem a diferentes significados do mesmo termo ou conceito (sinonímia), constituindo um nó da rede. Como exemplo, temos os termos “automóvel” e “carro”, que estão inseridos no mesmo synsets, já que ambos são a lexicalização do mesmo conceito. A figura 8 apresenta uma ilustração básica de semelhança e relação semântica encontrada pelo WordNet.

Figura 14: Relacionamento entre conceitos no WordNet



Fonte: autor

O WordNet tem sido usado com sucesso em muitas atividades de PLN. Suas relações semânticas criam uma estrutura hierárquica muito útil para a linguística computacional e o processamento de linguagem natural (Meng et al., 2013).

3.4 USO DE TESAUROS EM ENGENHARIA DE SOFTWARE

Segundo Wazlawick (2013), a busca por artefatos de software durante a manutenção ou evolução do software está presente em diversas atividades da Engenharia de Software. Tarefas comuns incluem busca automática em repositórios

de relatórios de erros, código para reutilização, solicitações de mudança e casos de teste. Para Rahman & Roy (2018), nem sempre é fácil localizar documentos de interesse em grandes repositórios, devido a incompatibilidades de vocabulário. O uso de termos concatenados, abreviações e acrônimos para representar uma funcionalidade ou conceito existente dificulta a recuperação de documentos relevantes, forçando muitas vezes a reformulação das consultas.

Como exemplo no domínio de para dispositivos móveis, o termo "Fingerprint" que remete a uma funcionalidade dos aparelhos, pode sofrer variações como “finger print”, “finger”, “FP”, entre outros. Tais variações dificultam a recuperação automática de artefatos relevantes referentes a esse componente, pois os motores de busca convencionais não associam a palavra consultada a variações do termo original.

Nesse contexto, uma abordagem promissora para aumentar as chances de recuperação de documentos relevantes é a expansão de consultas baseadas em tesauros. Ainda assim, o exemplo acima deixa claro que, no caso de recuperação de artefatos de SW, a melhor opção é usar um tesauro de domínio específico, uma vez que um tesauro genérico não conterá tais variações.

3.6 CONSIDERAÇÕES FINAIS

Os tesauros são instrumentos poderosos que agregam significativo valor a sistemas de RI, permitindo definir os termos que devem ou não fazer parte do vocabulário controlado. Em domínios específicos que envolvem linguagem natural, tesauros oferecem um vocabulário restrito para aquele domínio.

É importante enfatizar que a escolha da abordagem para a construção de tesauros deve levar em consideração os aspectos do domínio de aplicação. A criação automática de tesauros tem demonstrado ser bastante eficaz.

4 UM PROCESSO PARA CONSTRUÇÃO DE TESAUROS DE DOMÍNIO ESPECÍFICO

Este capítulo descreve o trabalho realizado, contando com 5 seções: 4.1 descreve como o tesauro pode auxiliar no processo de teste de software; 4.2 apresenta a visão geral do processo, mostrando suas etapas; 4.3 apresenta as bases de dados utilizadas no processo; 4.4 descreve as etapas de processamento; a seção 4.5 apresenta brevemente trabalhos relacionados e a seção 4.6 traz as considerações finais.

4.1 TESAUROS PARA AUXILIAR PROCESSOS DE TESTE DE SW

Durante as fases de desenvolvimento e manutenção do software, as atividades de teste são centrais para garantir a qualidade dos produtos. Segundo Myers et al. (2011), o teste de SW consiste em um conjunto de processos que visam garantir que aplicações e sistemas funcionem como planejado, evitando surpresas aos usuários finais. Software que não funciona corretamente pode levar a muitos problemas, incluindo a perda de tempo, dinheiro e reputação comercial.

Testar o software é essencial para identificar desvios que possam impactar o resultado esperado, e corrigi-los antes do lançamento do produto. O processo de teste possui diversas fases, tais como: planejamento de teste, análise, modelagem e execução dos testes, tratamento das solicitações de mudanças, entre outras. Para garantir que os testes sejam realizados satisfatoriamente, as empresas de desenvolvimento de software estão investindo em equipes de teste qualificadas e em novas metodologias para aprimorar as atividades de teste (Rajani, 2012).

As empresas de software, em geral, não adotam regras ou diretrizes sobre como nomear termos específicos de domínio (ou seja, termos técnicos relacionados às funcionalidades do software). Embora os funcionários conheçam os termos por completo, o uso de abreviações, acrônimos e concatenações arbitrárias para se referir à mesma entidade é algo comum dentro de empresas de SW. Segundo Abreu (2010), a adaptação da linguagem de comunicação falada por pessoas ligadas ao mundo digital é frequentemente usada para economizar tempo na escrita.

No contexto de teste de SW, o tesauro pode ser útil para várias tarefas onde é necessário selecionar automaticamente documentos relevantes a partir de bases indexadas. Como exemplos, citamos a criação de Planos de Teste apoiada por bases indexadas de Casos de Teste, e a criação e recuperação de Solicitações de Mudança do SW relacionadas a uma dada versão do SW. A expansão automática das consultas a essas bases indexadas, adicionando termos relacionados aos seus termos originais (sinônimos, abreviaturas, neologismos, termos concatenados, entre outros) tende a recuperar mais documentos relevantes, melhorando assim a cobertura dos sistemas de RI que utilizam buscas por palavras-chaves.

No domínio de sistemas mobile, é possível encontrar diferentes variações terminológicas para nomear a mesma entidade. Como exemplo, citamos o recurso “fingerprint sensor”, funcionalidade presente em diversos dispositivos. Esse recurso recebe várias designações, como: “finger print”, “finger”, “FP”, “FPS”, etc. Neste caso, como em muitos outros, o tesauro tem papel essencial como meio de agrupamento de termos relacionados entre si, podendo ser armazenado de maneira simples em uma base de conhecimento.

4.1.1 Contexto do trabalho

A Motorola Mobility mantém diversas iniciativas que visam melhorar, agilizar e reduzir custos de todo o processo de teste. Destacamos as iniciativas centradas no uso de repositórios de artefatos de SW, como documentos de solicitações de mudança (do inglês, *Change Request* - CR) e Casos de Testes (CT). Cada um desses repositórios é indexado e acessado por meio de sistemas individuais que incorporam um mecanismo de busca e recuperação de informação. As bases de documentos são estruturadas como um arquivo invertido, no qual cada documento é indexado pelas palavras que ele contém (Baeza-Yates & Ribeiro-Neto, 2011).

O sistema *Auto Test Plan* - ATP (Magalhães, 2020), que indexa CTs, é utilizado como apoio na construção automática de Planos de Teste. A partir do texto encontrado nas Notas de Versão (*Release Notes*), ATP recupera CTs relacionados aos erros que foram corrigidos na versão atual do dispositivo sendo testado (ver seção 5.2.2). Já o sistema AVS (*Automatic Versatile Search tool*) (Santos et al., 2018) recebe como entrada descrições de erros encontrados pelo testador, e retorna

CRs relacionadas já existentes na base indexada. Seu objetivo é evitar a criação de CRs duplicadas (seção 5.2.1).

Esses sistemas usam consultas baseadas em palavras-chave para recuperar itens relevantes, retornando documentos cujos termos correspondem às palavras da consulta. No entanto, devido às variações de terminologia existentes, os termos usados na consulta nem sempre estão presentes nos documentos relevantes. Por exemplo, uma consulta como o termo "finger print" não é capaz de recuperar documentos que usem termos equivalentes, como "fingerprint", "FP", ou "FP Sensor". Assim, a consulta não irá recuperar todos os documentos relevantes na base, a fim de garantir a finalização adequada da atividade de teste atual.

Os efeitos negativos da variação de terminologia na empresa podem ser minorados através do uso de um tesauro acoplado aos sistemas existentes. Contudo, a Motorola Mobility não conta com um dicionário de sinônimos de domínio específico para uso generalizado. Os tesauros genéricos (e.g., WordNet) não são adequados para suas necessidades particulares, por duas razões principais: ambiguidade lexical (palavras com dois ou mais significados diferentes); e por não cobrirem abreviações e acrônimos específicos usados nos documentos da Empresa.

Diante disso, constatamos a necessidade de uma solução computacional capaz de automatizar, com o mínimo de interferência humana, a construção de um tesauro específico para a empresa. As seções a seguir apresentam o processo proposto, e o Capítulo 5 traz detalhes sobre a implementação do protótipo atual. O tesauro construído como resultado deste trabalho de mestrado foi acoplado aos sistemas ATP e AVS, expandindo as consultas originais com termos relacionados. Os resultados obtidos foram muito positivos (ver seção 5.2.2.1 e seção 5.2.1.1).

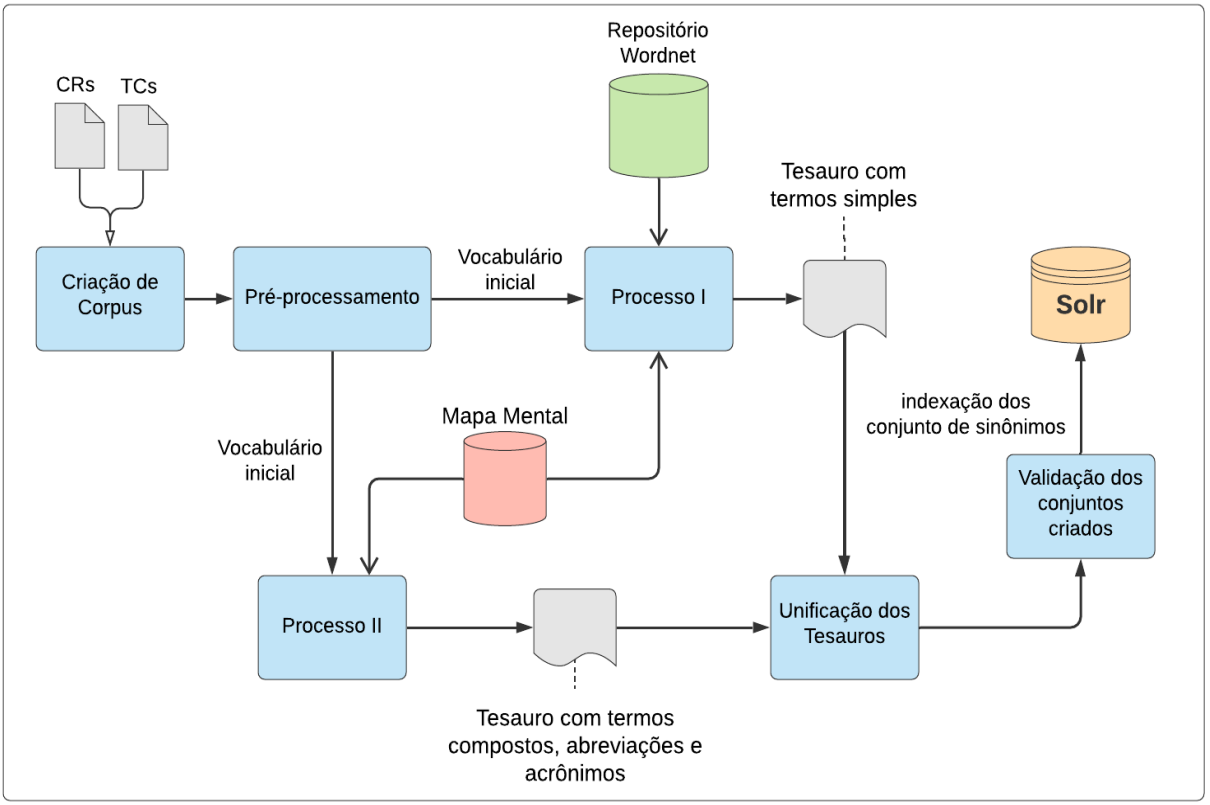
4.2 VISÃO GERAL DO PROCESSO

Com o intuito de contribuir para a melhoria de atividades variadas realizadas em empresas de SW, propomos aqui um processo de criação de tesauros de domínio específico a partir de documentos internos da empresa. O processo recebe como entrada documentos locais e retorna synsets (conjuntos de termos relacionados). Dessa forma, os synsets conterão apenas palavras/termos específicos do domínio, evitando ambiguidade lexical. Assim, o tesauro pode ser

usado com segurança para tarefas variadas que envolvem processamento de texto, com destaque para expansão de consultas em sistemas de RI, com o objetivo de melhorar a cobertura desses sistemas sem comprometer sua precisão.

A figura 15 apresenta o processo geral do protótipo, considerando as bases de dados utilizadas e os principais módulos de processamento. Como mostra a figura 9, o processo de criação do tesauro se apoia em duas bases de dados/conhecimento que relacionam palavras e termos compostos (seção 4.3). A partir dessas bases e dos documentos de entrada, as etapas de processamento são executadas, resultando em uma lista de conjuntos de palavras relacionadas dentro do domínio escolhido (seção 4.4). Essa lista pode ser indexada via um sistema de RI, ficando disponível para acesso manual ou automático.

Figura 15: Diagrama de fluxo de dados do processo



Fonte: autor (2021)

Esse processo foi implementado em um sistema protótipo denominado “Motosaurus”, utilizando métodos e técnicas de Mineração de Dados, Recuperação de Informação e Processamento de Linguagem Natural.

Como já mencionado, o tesauro criado foi usado inicialmente na expansão de consultas em dois sistemas de RI bastante utilizados dentro da empresa, o ATP e o AVS. É importante mencionar que o tesauro também foi utilizado em outros sistemas da empresa, como por exemplo um sistema de classificação de casos de teste automatizáveis (Matos, 2020). Porém, o desempenho desse sistema não foi avaliado em detalhes considerando a presença/ausência do tesauro, portanto ele não será apresentado neste documento. Vale ressaltar ainda que o tesauro poderá ser facilmente recriado quando houver necessidade, mantendo-se sempre atualizado em relação aos novos termos usados na empresa.

A seção 4.3 apresenta as bases de dados utilizadas no processo e a seção 4.4 detalha os módulos de processamento. O protótipo atual está implementado na linguagem de programação Python. Na seção 4.4 serão descritas com mais detalhes todas as etapas do processo proposto.

4.3 BASES DE DADOS UTILIZADAS NO PROCESSO

O processo proposto utiliza duas bases de dados lexicais para apoiar a criação do tesauro específico: o WordNet e o Mapa Mental. Como o WordNet já foi apresentado na Seção 3.3.2.2, vamos descrever aqui apenas o Mapa Mental.

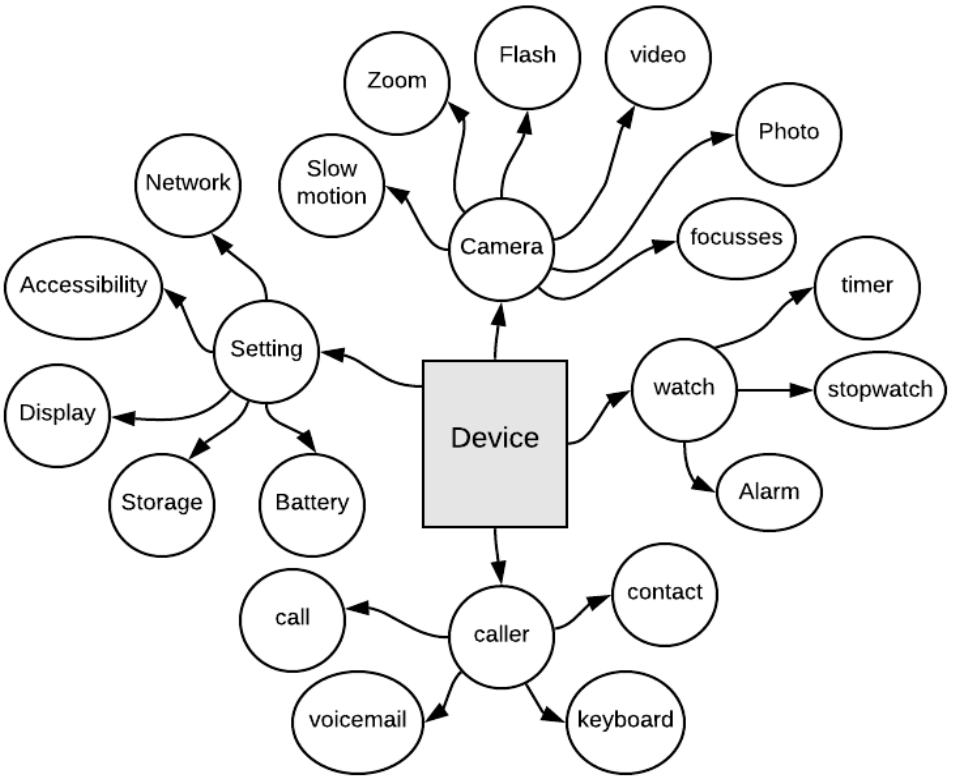
4.3.1 Mapa Mental

O Mapa Mental foi construído manualmente com base na estrutura do domínio em foco (dispositivos móveis). Cada nó (semente) representa uma funcionalidade específica desses aparelhos e os links representam as maneiras possíveis de navegar de uma funcionalidade para outra. Vejamos o exemplo de câmera, que é uma funcionalidade bastante comum nos dispositivos móveis. A primeira semente, “camera”, é interligada a outras cinco sementes (“slow motion”, ”zoom”, “flash”, “video”, “photo” e “focusses”).

Como dito, esse arquivo foi criado com base na estrutura do domínio em foco. As funcionalidades (nós) e suas relações foram determinadas com base em estudos realizados sobre os dispositivos móveis da empresa. Os nomes dos nós foram escolhidos a partir da análise dos documentos internos, com base na terminologia

usada dentro da empresa. A figura 16 apresenta um trecho do Mapa Mental com algumas sementes que foram usadas neste trabalho.

Figura 16: Trecho do Mapa Mental



Fonte: autor (2021)

Os nós (sementes) são usados pelo Processos I de geração do tesauro, guiando a escolha do “termo descritor” (cabeça) do conjunto inicial de sinônimos (com termos simples/palavras isoladas). No Processo II, esses conjuntos iniciais serão enriquecidos com termos compostos, abreviações e acrônimos, que são gerados a partir das mesmas sementes do seu nó descritor.

Os links não foram usados nesta versão do protótipo, porém, eles são úteis para verificar relações de especialização/generalização entre funcionalidades. Por exemplo, podemos criar um subconjunto de termos relacionados a partir de um nó de mais alto nível. A funcionalidade camera pode dar origem ao seguinte subconjunto de termos: Camera [slow motion, zoom, flash, vídeo, photo, focusses].

O Mapa Mental atual conta com 195 nós e 204 relações entre eles. Essa base de dados pode ser facilmente alterada/atualizada, quando houver necessidade de inclusão ou exclusão de novas funcionalidades ou relações. O processo manual de

criação desta base de dados foi relativamente rápido e simples (em média, 5 dias de trabalho), uma vez que nós já tínhamos conhecimento aprofundado sobre esse domínio. A escolha dos nomes das sementes também foi bastante simples, devido ao nosso conhecimento prévio sobre a terminologia adotada na empresa.

Atualmente, o Mapa Mental é criado de forma manual, como já mencionado. Não investimos tempo na automatização desse processo porque o trabalho de construção é simples e rápido, e esse mapa não requer atualizações frequentes. Essa (semi) automatização pode ser indicada como trabalhos futuros. Porém, é importante salientar que o Mapa Mental é intrinsecamente dependente do domínio. Assim, a sua criação sempre irá depender de um especialista no domínio. Uma sugestão seria apenas criar uma interface de uso simples para o especialista apenas informar os nomes das entidades e suas relações. Uma solução totalmente automática dependeria de informações de domínio já coletadas e armazenadas em um formato reusável (por exemplo, na forma de uma ontologia).

4.4 ETAPAS DE PROCESSAMENTO

Como mostra a figura 9, o processo geral de criação do tesauro conta com seis módulos: Criação do Corpus; Pré-processamento; Processo I (termos simples); Processo II (criação de termos compostos, abreviações e acrônimos); Unificação dos tesauros construídos nos processos I e II; e Validação dos conjuntos criados. As seções a seguir detalham cada uma dessas etapas.

4.4.1 Criação do corpus de documentos

Empresas de SW mantém bases de documentos textuais, armazenando os principais artefatos produzidos durante o ciclo de vida do SW. No contexto de teste de software, é possível encontrar um grande volume de artefatos que descrevem casos de teste e solicitações de mudança, que, em geral, são armazenados junto com a respectiva versão do software à qual eles estão relacionados.

A primeira etapa do processo de criação do tesauro específico consiste em coletar documentos textuais contidos nos repositórios da empresa, a fim de criar o corpus de documentos a partir do qual o tesauro será gerado. Como já mencionado, o protótipo atual utiliza CTs e CRs como documentos de entrada. Porém, é

importante ressaltar que o tesauro pode ser gerado a partir de documentos diversos, com algumas adaptações do código que lê os documentos de entrada. Além disso, ele poderá ser recriado sempre que houver necessidade, mantendo-se sempre atualizado em relação aos novos termos usados na empresa.

Por meio de uma análise manual feita nos CTs e CRs coletados, foram identificados campos com informações diversas que poderiam ser extraídas, por exemplo: id, titulo, resumo, descrição, data, entre outros. No entanto, nem todas essas informações são relevantes para nossa tarefa. Após uma breve análise da utilidade e relevância das informações disponíveis, foi criado um script para extrair dos documentos apenas os campos com informações textuais (título, resumo e descrição). Os campos extraídos de todos os documentos são concatenados, criando uma espécie de “bag of words”, porém ainda contendo caracteres especiais. Essa lista é passada para a etapa de pré-processamento.

4.4.2 Módulo de Pré-processamento

Nesta etapa, vários procedimentos de normalização textual dos dados de entrada são executados automaticamente. O objetivo é eliminar caracteres, termos e outros dados indesejados, a fim de restringir o conjunto de termos antes de enviar aos algoritmos de criação de dicionário de sinônimo.

Esse módulo utiliza scripts na linguagem de programação Python, juntamente com a biblioteca NLTK[15], que utiliza técnicas de PLN. Como os documentos da empresa estão todos em inglês, foram utilizadas bibliotecas voltadas para a língua inglesa. Após a tokenização, itens irrelevantes (links, dígitos, caracteres especiais, etc) são removidos através de regras baseadas em expressões regulares. Essas regras foram criadas usando a biblioteca re NLTK[16].

Um PoS-tagger realiza a etiquetagem morfossintática dos termos restantes. Apenas termos pertencentes às classes gramaticais de substantivos e verbos foram considerados nessa etapa, uma vez que não observamos relevância de termos de outras classes gramaticais na composição do tesauro específico. Outra técnica

[15] https://www.nltk.org/
[16] https://www.nltk.org/re

utilizada no pré-processamento é a remoção de stopwords, para diminuir a dimensionalidade do conteúdo e melhorar a relevância dos dados.

As palavras selecionadas nesta etapa darão origem ao Vocabulário Inicial da base de documentos, utilizado pelos processos subsequentes de criação do tesauro.

4.4.3 Criação do tesauro com palavras isoladas (Processo I)

Esta etapa recebe a lista de palavras que resultaram da etapa de pré-processamento (vocabulário inicial) e cria a primeira versão dos synsets (conjuntos de sinônimos), que conta apenas com palavras isoladas. Esse processo se apoia em consultas ao WordNet, como veremos a seguir. Vale relembrar que a empresa não utiliza o WordNet como tesauro porque seus synsets são genéricos, contendo (quase) todos os termos do idioma inglês, nós estamos interessados em termos pertencentes ao domínio da empresa.

No contexto deste trabalho, o WordNet foi usado para agrupar palavras relacionadas que ocorrem no nosso corpus. Cada termo oriundo do Vocabulário Inicial é submetido ao WordNet, que retorna as palavras com significado semelhante ao termo de entrada. Para isso, é utilizada uma API em Python que permite ter acesso ao dicionário de sinônimos disponível na plataforma do WordNet.

Os termos retornados pelo WordNet que também pertencem ao Vocabulário Inicial serão selecionados, dando origem a um conjunto de termos relacionados. Assim, teremos vários conjuntos de termos agregados a partir dessas consultas ao WordNet, que passarão por outros processos de refinamento.

Para cada termo k no Vocabulário inicial, faça:

(i) Envie k para WordNet como uma consulta;

(ii) Verifique a ocorrência de cada termo retornado do WordNet no Vocabulário inicial;

(iii) Adicionar ao conjunto de sinônimos do termo k apenas os termos retornados pelo WordNet que também estão presentes no vocabulário inicial.

Este procedimento garante que os conjuntos de sinônimos do tesauro conterão apenas palavras do domínio da empresa. Por exemplo, o termo "picture" recupera termos do WordNet como "image", "painting", "scene", "photo", etc. No

entanto, apenas os termos "photo" e "image" serão incluídos no synsets de "picture", uma vez que são os únicos que ocorrem no vocabulário inicial. Neste ponto, podemos ter redundâncias nos conjuntos de sinônimos. Por exemplo, considere as seguintes palavras do vocabulário inicial: "image", "picture" e "photo". Os synsets obtidos a partir das consultas ao WordNet serão:

- image [picture, icon]
- picture [photo, image, exposure, icon]
- photo [photography, exposure, picture, pic]

Todos os conjuntos citados acima estão relacionados à mesma entidade de domínio, "photo". Então, a seguir é feita uma concatenação desses conjuntos em um único synset, considerando apenas as palavras que ocorrem no vocabulário inicial. Assim, o conjunto de final de sinônimos vai conter apenas as seguintes palavras: [image, photo, picture, pic].

Em seguida, é definido o descritor desse conjunto de sinônimos, que é escolhido com base nas palavras que aparecem no MindMap. No exemplo acima, a palavra “photo” se tornará o descritor do conjunto, resultado no synset final abaixo:

- photo [image, picture, pic]

4.4.4 Derivação de Termos Compostos (Processo II)

O processo anterior constrói conjuntos de sinônimos apenas com palavras simples. No entanto, o vocabulário do domínio também contém termos compostos (com duas ou mais palavras) que se referem a uma entidade de domínio única (por exemplo, "lock screen"). Além dos termos compostos, também é possível encontrar nos documentos variações dos termos de domínio, como abreviações, acrônimos e concatenações livres de termos compostos. Todas essas variações devem fazer parte do tesauro, uma vez que são consideradas como termos relacionados ou sinônimos. Esses termos podem ser substantivos que apontam para uma determinada funcionalidade ou componente do aparelho.

Para obter essas variações, foram utilizadas diferentes técnicas, como: marcação de classe gramatical, utilização de regras de expressões regulares e formação de n-gramas. Além disso, uma medida de similaridade baseada na

distância de edição é usada para comparar as variações construídas com os termos que aparecem no vocabulário inicial. Por fim, as sementes do Mapa Mental são utilizadas para uma organização final desses conjuntos. Note que esses termos são de grande importância na expansão de consultas a sistemas de RI, a fim de garantir uma boa taxa de cobertura.

4.4.4.1 Criação de Bigramas

Esta etapa produz uma representação mais sofisticada de tokens com base em bigramas. O processo concatena dois substantivos contíguos ou substantivo seguido de verbo no vocabulário inicial, formando um novo termo composto. A representação de bigrama pode ser obtida da seguinte forma:

- **Lista de tokens do vocabulário inicial:**
  [“finger”, “print”, “camera”, ”app”, “lock”, “screen”, “advance”, “calling”]
- **Representação em forma de bigrama:**
  [“finger print”, “camera app”, “lock screen”, “advance calling”]

  Os termos compostos são utilizados pela próxima etapa de processamento.

4.4.4.2 Criação de Concatenações, Abreviações e Acrônimos

Esta etapa recebe a lista de termos compostos criados na etapa anterior e gera uma lista de abreviações, acrônimos e concatenações relacionadas. Esta etapa usa scripts RegEx como suporte ao processo de criação. As seguintes fases de processamento são realizadas aqui:

- **Concatenação:**
  Advance Calling -> [AdvanceCalling, Advance-Calling]
- **Criação de abreviações: (usando RegEx)**
  Advance Calling -> [AdvCalling, AdvanceCall]
- **Criação de acrônimos:**
  Advance Calling -> AC

A etapa de criação de abreviações realizado no Processo II, também é executada com conjuntos de palavras únicas oriundos do Processo I, criando abreviações como "cam" para "câmera".

Todos os termos criados pela etapa acima são validados em relação ao vocabulário inicial, e apenas os termos que ocorrem com no mínimo 70% de similaridade com termos do vocabulário inicial serão incluídos no conjunto de sinônimos. Para isso, utilizamos uma medida de similaridade que calcula a distância de edição entre strings (Levenshtein, 1966).

Os termos aprovados nesta validação são incluídos nos respectivos conjuntos de termos relacionados. No exemplo citado acima, os tokens "Advance-Calling" e "AC" foram descartados porque não atingiram uma similaridade mínima com termos existentes no vocabulário inicial, ou simplesmente não aparecem nesse vocabulário. Assim, o exemplo acima contribuiria com os seguintes termos:

- Advance Calling [AdvanceCalling, AdvCalling, Advance-Call]

Os synsets que foram aprovados no Processo II passam para uma última etapa, que é a definição do descritor do conjunto. Assim como no Processo I, essa escolha é baseada nas palavras que aparecem no Mapa Mental.

Por fim, é importante esclarecer aqui que os synsets que não possuem nenhum termo presente entre as sementes do Mapa Mental são descartados, tanto no Processo I quanto no Processo II.

4.4.5 Unificação dos synsets criados

Os synsets criados nos Processos I e II devem ser unificados e organizados de maneira adequada para então gerar o tesauro final. Nesta etapa, todos os conjuntos de sinônimos criados pelos processos anteriores são mesclados. Os conjuntos de sinônimos relacionados à mesma entidade de domínio se transformam em um único synset. Por exemplo, o conjunto unificado para a entidade de domínio "câmera" será:

- **Synset construído no processo I**
  camera [camcorder]

- **Synset construído no processo II**
  camera [cam, camera app, camapp,camera1, cam1, camera2, cam2]

- **Unificação dos synsets:**
  camera [cam, camera app, camapp, camcorder, camera1, cam1, camera2, cam2]

Vale salientar que cada termo do vocabulário só pode fazer parte de um synset. Ou seja, o mesmo termo não pode aparecer em dois ou mais conjuntos de termos no tesauro ao mesmo tempo. Essa decisão busca eliminar qualquer tipo de ambiguidade lexical causada por palavras iguais que poderiam estar associadas a mais de um significado.

4.4.6 Validação dos synsets criados

Apesar de todos os cuidados para gerar corretamente o tesauro, ainda é necessário realizar uma inspeção manual para verificar se existe alguma anormalidade nos conjuntos gerados. Nesta etapa, uma validação manual é realizada por um usuário que conhece bem os documentos e processos da empresa. Apenas os conjuntos que sobreviveram aos processos anteriores são contemplados nesta etapa. O objetivo é verificar a possibilidade de remover ou inserir manualmente termos dos conjuntos criados.

Para simplificar a validação, foi desenvolvida uma interface de fácil interação que provê ao usuário meios de modificar os conjuntos de sinônimos conforme necessário. Por meio dessa interface, o processo de validação se torna rápido e fácil, sendo realizado apenas uma vez a cada dicionário de sinônimos criado.

A validação realizada por um analista com conhecimento do domínio apresentou bons resultados. O tempo de verificação dos conjuntos de sinônimos não ultrapassou cinco minutos, e o grau de assertividade, segundo o analista, foi muito bom. Não foi possível realizar validação dos conjuntos com mais de um analista, uma vez que a empresa parceira tem uma alta demanda de trabalho. Com isso, tornou-se difícil conseguir outros analistas para realizar a validação em duplicidade.

4.4.7 Indexação do Tesauro criado

Uma vez que os conjuntos de sinônimos foram criados e validados, faz-se necessário utilizar uma ferramenta de indexação que facilite a busca por esses synsets sempre que desejado. Para isso, foi utilizado o Apache Solr, já mencionado anteriormente. A ferramenta de indexação permite buscas simples ou mais refinadas, através do uso de metadados.

Os synsets foram organizados em uma estrutura JSON, para uma melhor utilização pela ferramenta Solr. Para isso, foi utilizada a biblioteca *pysolr*[17] para Python, que permite estruturar o documento da maneira desejada. Para o trabalho proposto, cada synset possui dois campos, sendo um campo o ID (representado pelo descritor do synset), e outro contendo a lista de termos relacionados. Esses campos são identificados através da definição de metadados (id e contente).

```
{
“id”: “fingerprint”,
“content”: [ "finger print",  "fingerprint sensor", "finger", "fp  ]
}
```

O Sorl possui um recurso através do qual é possível definir a maneira como a busca é realizada. Tais configurações ocorrem dentro de uma estrutura chamada CORE. O CORE precisa ser definido antes que a indexação aconteça. Também é possível configurar o CORE no arquivo *managedschema.xml* para realizar um pré-processamento dos arquivos indexados, bem como das queries submetidas ao sistema. Em nosso caso, não usamos pré-processamento, apenas o recurso de definição de campos com metadados. O script responsável pela preparação dos dados a serem indexados organiza uma estrutura de maneira que cada ID do synset seja único. Ou seja, não existem synset com ID iguais.

Assim, o tesauro final torna-se disponível para consulta manual (via interface) ou para consultas automáticas, através da sua integração a sistemas variados (como já mencionado no caso dos sistemas ATP e AVS).

[17] https://github.com/django-haystack/pysolr

4.5 TRABALHOS RELACIONADOS

Nesta seção, são descritos alguns trabalhos relacionados. Como não foram encontrados métodos de construção de tesauro diretamente ligados à mesma área do trabalho proposto, selecionamos algumas abordagens de outras áreas, porém com objetivos similares.

4.5.1 Um método para criação de tesauros semânticos de requisitos usando a ontologia SKOS

Em (Zacarias, Jacyntho & Vasconcelos, 2020), os autores definem um método para criação de tesauro usando Sistemas de Organização do Conhecimento (SOC) com tecnologias da Web Semântica. O objetivo é fornecer um conjunto de regras voltadas à criação de tesauros que auxiliem no reuso de requisitos em projetos de software, criando uma estrutura que representa as descrições dos requisitos com anotações semânticas que possam ser utilizadas para beneficiar atividades de recuperação de requisitos em futuros projetos de software.

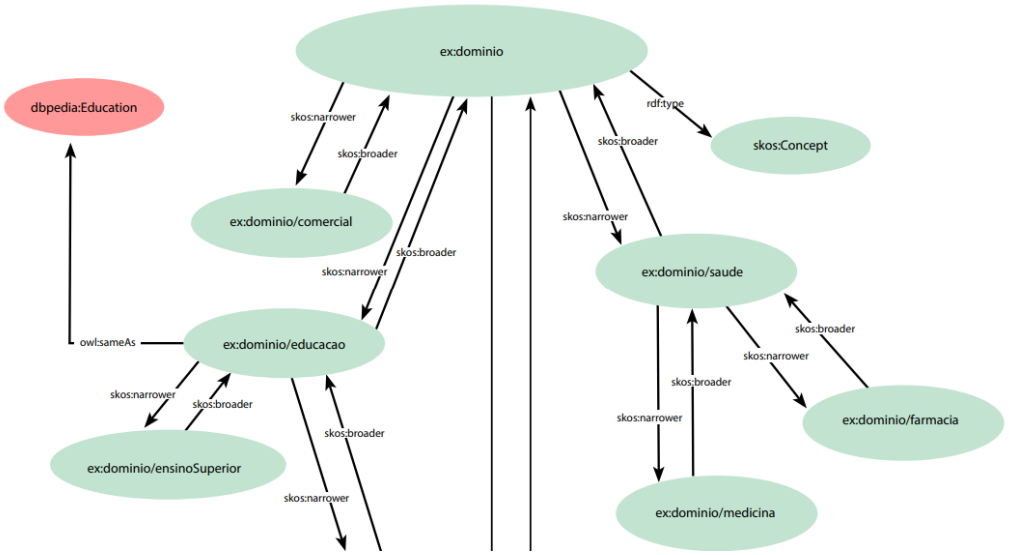
Os sistemas SOC, segundo Altounian (2019), podem ser definidos como uma estrutura que ajuda na representação, organização e gestão do conhecimento. Por exemplo, vocabulários controlados, taxonomias e tesauros. O autor ainda destaca que o tesauro é um SOC que classifica e organiza conceitos em forma de hierarquia equivalente e associativa.

A Web semântica, segundo Berners-Lee, Hendler & Lassila (2001), destaca-se por atrelar significado às informações. Com isso, ao associar documentos publicados em estrutura de metadados, seus conteúdos podem ser processados para serem utilizados em processos de tomada de decisão.

Com a unificação de SOC e Web semântica, os autores descrevem um método para representação sistemática de requisitos em um sistema por meio de conceitos em tesauros SKOS (Simple Knowledge Organization System). Os benefícios são uma maior compreensão, devido à semântica explicativa. Com isso, os stakeholders do projeto com pouco conhecimento do domínio poderão reusar essa estrutura e, consequentemente, haverá uma redução de tempo e esforços

empregados na fase de elicitação de requisitos. Na figura 15 é mostrado um trecho do grafo RDF do tesauro de domínio de Conhecimento/Negócios.

Figura 17: Trecho do grafo RDF do tesauro de domínio de Conhecimento/Negócios.



Fonte: (Zacarias, Jacyntho & Vasconcelos, 2020)

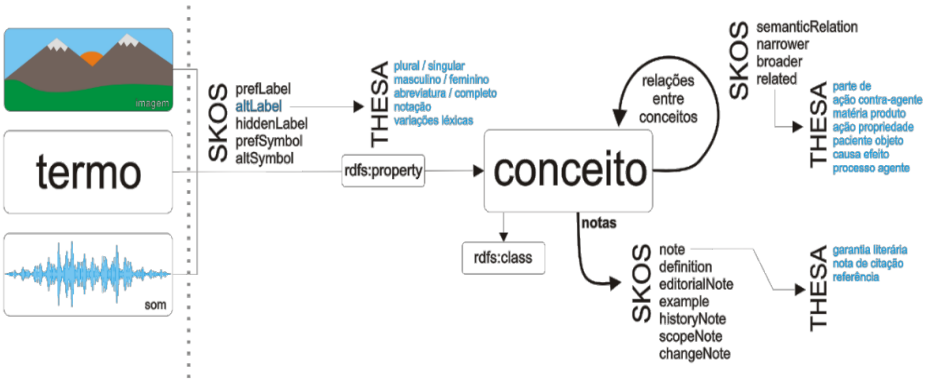
4.5.2 THESA: ferramenta para construção de tesauro semântico aplicado interoperável

Em Gabriel Junior & Laipelt (2017), é descrito o THESA, software para elaboração de tesauros semânticos que podem ser usado pelos estudantes de biblioteconomia na disciplina de linguagem documentária. A ferramenta foi desenvolvida na linguagem de programação *php* juntamente com o banco de dados *MySql,* de forma a possibilitar o compartilhamento e desenvolvimento da ferramenta. O Thesa permite que os usuários possam criar múltiplos tesauros em diferentes áreas do conhecimento, seguindo uma concepção de relação entre conceitos, partindo do pressuposto de que um conceito pode ser representado por um termo, uma imagem, um som, um link ou algo que possa ser explicitado.

Para nortear as ligações entre conceitos, o THESA elabora grafos que são utilizados para visualização das relações existentes, gerando automaticamente o mapa conceitual das relações entre termos. Além disso, a ferramenta incorpora especificidades como “parte de”, “causa efeito”, “ação propriedade” entre outras para

gerar relações semânticas entre conceitos. A figura x mostra a estrutura da ferramenta THESA.

Figura 18: Modelo da estrutura do Thesa



Fonte: Gabriel Junior & Laipelt (2017)

Os descritores incorporam algumas descrições que representam relações entre o conceito e o termo, como singular, gênero, abreviação, notação e variações léxicas, e o grau de semântica entre termos é representado conforme o nível de interoperabilidade que precise do SOC ou do SRI.

4.6 CONSIDERAÇÕES FINAIS

Neste capítulo, foi apresentado o processo de construção de tesauro para um domínio específico. A combinação das técnicas utilizadas nos Processos I e II foi crucial para obter resultados satisfatórios. Os tesauros são construídos manualmente quando cada ferramenta é lançada, não sendo compartilhados com outras ferramentas e colaboradores, tornando-se obsoletos por falta de atualização.

A solução proposta teve como meta principal oferecer um ambiente para que os colaboradores pudessem criar, consultar e atualizar de maneira fácil e rápida as listas de sinônimos que são utilizadas dentro da empresa. Por meio de nosso processo, o tempo gasto para criar um novo dicionário, ou atualizar o existente, será reduzido consideravelmente.

Por fim, vale salientar que nosso processo conta com apenas uma etapa manual (Validação). No capítulo de conclusão, serão indicados trabalhos futuros que visam mitigar essa pendência.

5 MOTOSAURUS - ESTUDO DE CASO

Este capítulo apresenta o Motosaurus, o protótipo desenvolvido para auxiliar na construção de tesauros de domínio específico (seção 5.1). Este estudo de caso teve como foco a área de testes de software para o domínio de dispositivos móveis.

Durante o desenvolvimento do protótipo, foi construído um tesauro inicial a partir de um corpus de aproximadamente 1.000 CTs e 1.000 CRs. O processo de construção seguiu todos os passos descritos na Figura 9 (Capítulo 4). A etapa de Validação (manual) foi realizada por um funcionário experiente, que gastou apenas 1h30 nessa atividade, utilizando uma interface amigável.

Esse tesauro foi indexado via Solr, ficando disponível para consultas através da interface do usuário (seção 5.1). Além disso, o tesauro foi integrado a dois sistemas da Motorola Mobility já mencionados: AVS e ATP (ver seções 5.2.1 e 5.2.2, respectivamente). Realizamos testes comparativos entre os sistemas originais e os sistemas utilizando o tesauro criado para expandir consultas, e os resultados obtidos comprovaram os benefícios do uso do tesauro nesse contexto.

Como dito, o trabalho foi desenvolvido no contexto de um projeto de pesquisa em teste de software. No entanto, o processo de criação de tesauros pode ser adotado com sucesso em outros domínios particulares, desde que siga os procedimentos descritos neste trabalho.

As seções a seguir trazem detalhes de uso do sistema protótipo (seção 5.1), bem como resultados dos testes realizados (seção 5.2).

5.1 MOTOSAURUS – UM TESAURO PARA O DOMÍNIO MOTOROLA

Como mencionado, o Motosaurus foi implementado com base no processo descrito no Capítulo 4. O sistema recebe como entrada documentos produzidos na empresa e retorna grupos de termos relacionados que ocorrem nesse corpus.

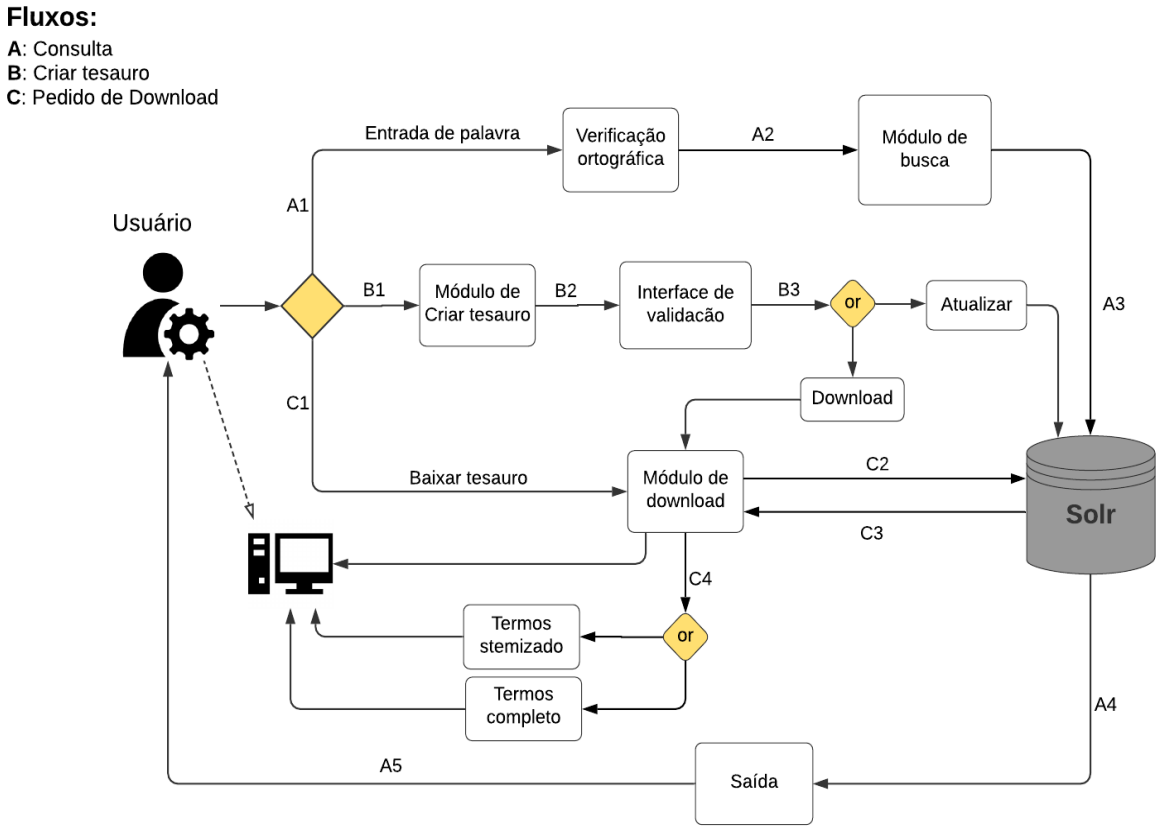
O Motosaurus é um protótipo que foi desenvolvido para ser uma aplicação Web, fazendo uso de tecnologias como Python, Flask[18] e AngularJS[19]. Sua interface pode ser usada tanto para criar novos tesauros como para realizar consultas no

[18] Microframework Python desenvolvimento de aplicações Web simples. Site: http://flask.pocoo.org/.
[19] Framework JavaScript para o desenvolvimento de aplicações Web. Site: https://angularjs.org/.

tesauro já existentes. Também é possível atualizar os conjuntos de sinônimos de um tesauro existente.

A interface oferece três fluxos: busca, criação de tesauro e pedido de download. A figura 17 mostra uma visão geral de uso do protótipo desenvolvido.

Figura 19: Diagrama com visão geral de uso do protótipo



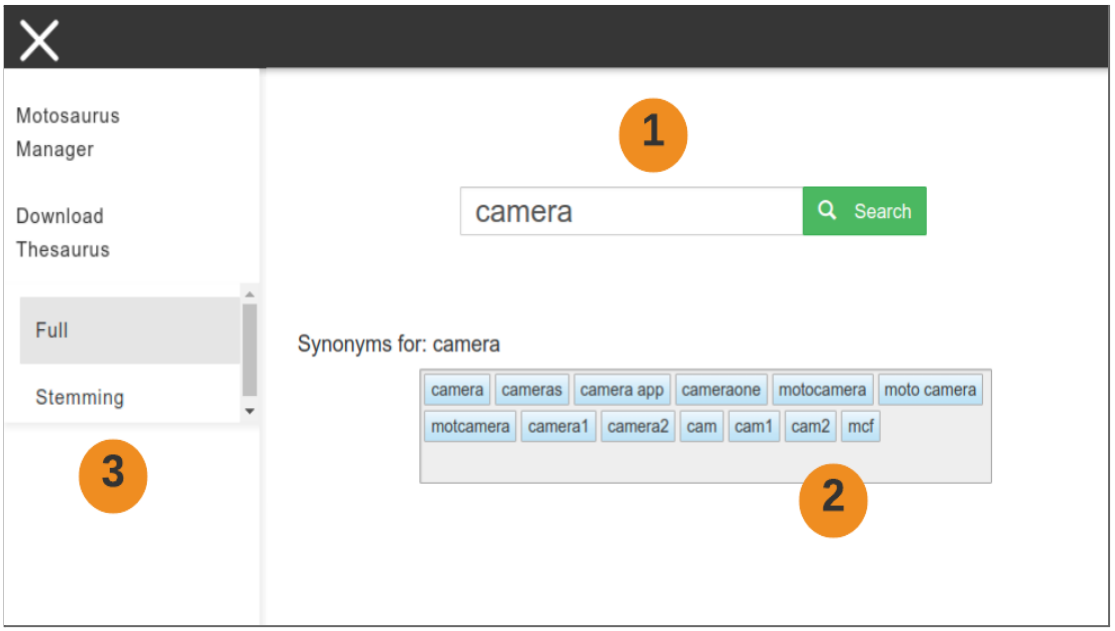
Fonte: autor (2021)

No primeiro fluxo (A), o protótipo oferece ao usuário a opção de realizar uma consulta ao Motosaurus. No segundo fluxo (B), é possível criar um novo tesauro a partir do input de um documento, ou simplesmente atualizar o tesauro já existente, que está salvo na base de dados. No terceiro fluxo (C), o usuário consegue fazer o download do tesauro que está salvo na base de dados. A seção 5.1.1 detalha as principais funcionalidades do protótipo.

5.1.1 Interface Principal do Motosaurus

A figura 12 apresenta um exemplo de uso da interface principal do sistema. Trata-se de uma consulta ao tesauro criado, a fim de recuperar termos relacionados à palavra “camera”.

Figura 20: Interface principal do sistema



Fonte: autor (2021)

(1) Campo de Pesquisa: Nesta janela, é possível digitar um termo a fim de recuperar seus sinônimos/termos relacionados existentes no tesauro. O campo de busca conta com um verificador ortográfico, para corrigir erros de ortografia na palavra digitada. As buscas também podem ser feitas por meio de uma API Rest [20] (conjunto de definições e protocolos usados no desenvolvimento e integração entre aplicações), que é disponibilizada pelo protótipo para integração do tesauro com outras ferramentas que necessitem expandir automaticamente suas consultas.

(2) Resultado da Busca: Essa janela exibe o conjunto de sinônimos que têm relação com o termo inserido no campo de pesquisa. Cada termo digitado retorna apenas um conjunto de sinônimos, e cada conjunto possui termos exclusivos. Ou seja, não existe o mesmo termo em dois ou mais conjuntos de termos no tesauro.

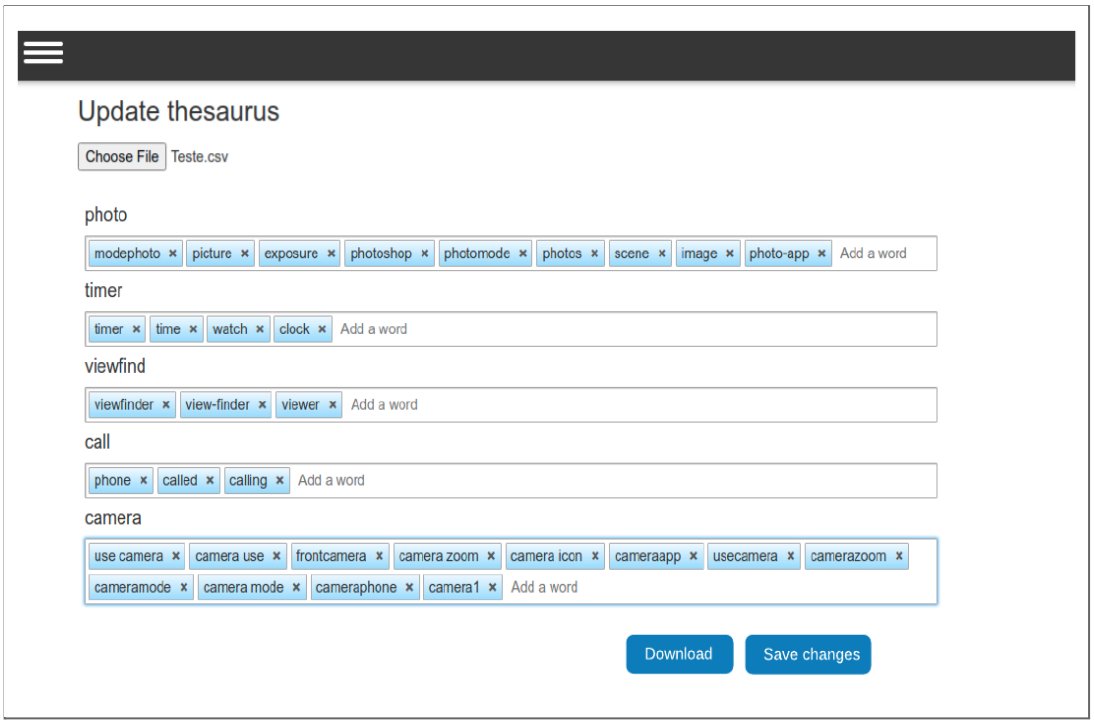
[20] https://pt.wikipedia.org/wiki/REST

(3) Menu Lateral: Nesta área são exibidas opções tais como gerenciar o tesauro existente e realizar download. A opção de gerenciar o tesauro é uma subpágina através da qual o usuário tem a possibilidade criar ou atualizar o tesauro. Através desse menu também é possível fazer o download do tesauro que está salvo na base de dados. O usuário tem duas opções: (1) com os termos completos, do modo como foram obtidos a partir do corpus de entrada; e (2) com os termos stemmizados (conservando apenas o radical das palavras). O formato do documento de download está de acordo com o que foi especificado pela equipe de suporte de desenvolvimento de SW da empresa.

5.1.2 Interface de Criação, Atualização e Validação

Na figura 13 é apresentada a interface através da qual o usuário pode inserir um novo corpus de documento e analisar os conjuntos gerados. Nessa figura, vemos que cada conjunto está separado em um campo de edição, sendo possível excluir ou inserir novos termos a qualquer conjunto existente. Assim, o processo de validação dos conjuntos automaticamente criados torna-se muito simples e rápido. Esse processo se faz necessário para evitar inconsistências no tesauro criado.

Figura 21: Ambiente de criação e atualização do tesauro



Fonte: autor (2021)

Na parte de baixo da tela, é possível ver as opções de baixar o tesauro gerado ou atualizar o tesauro existente. Essa última opção é essencial para evitar a obsolescência dos tesauros já criados. Depois que as atualizações são salvas, as próximas consultas e os pedidos de download são atualizados com os novos termos.

5.2 TESTES COMPARATIVOS

Como já mencionado, realizamos testes com dois sistemas já existentes, a fim de verificar os benefícios trazidos pelo uso do tesauro na expansão de consultas. Os dois sistemas, AVS e ATP, utilizam o Apache Solr como ferramenta de indexação. Assim, o tesauro foi inserido manualmente no próprio Sorl de cada sistema, e suas configurações foram devidamente ajustadas.

A seguir, veremos detalhes sobre esses sistemas, bem como os resultados dos testes comparativos. Os testes compararam, individualmente, o desempenho de cada sistema com e sem o uso do tesauro. Para prover uma análise mais justa, para cada sistema, foram utilizados os mesmos conjuntos de dados em ambas as rodadas. Cada sistema foi avaliado levando em consideração o seu contexto de construção e uso.

5.2.1 O Sistema AVS

Durante o processo de teste de software, algumas mudanças de código são necessárias para corrigir falhas encontradas ou para melhorar funcionalidades. Geralmente os erros encontrados são descritos em documentos de Solicitação de Mudanças (do inglês, *change request* – CR). Segundo Cavalcanti (2009), as CRs podem descrever situações de erros, solicitações de mudanças e melhoramentos, ou algum tipo de problema que possa ocorrer em um produto de software.

Para armazenar as CRs, as empresas de SW contam com sistemas gerenciadores que permitem que testadores, desenvolvedores, usuários e gerentes de projeto possam registrar e consultar o andamento das CRs submetidas (Nguyen et al., 2012). Como exemplos, citamos (Mantis, 2019), (Bugzilla, 2019) e Jira (Macieira, 2018). Como esses sistemas permitem sua utilização por diversos usuários simultaneamente, pode ocorrer a criação de várias CRs relacionadas à mesma situação de erro, gerando duplicidade de CRs. Esse problema causa

prejuízos às empresas, pois cada CR criada terá que ser analisada por um profissional e enviada à equipe de desenvolvimento, para correção do erro citado.

A maioria das abordagens adotadas para mitigar esse problema utiliza sistemas de busca e ranqueamento dos resultados computando a similaridade entre CRs. Nesse contexto, a Motorola Mobility vem investindo na detecção e mitigação de possíveis duplicações entre CRs antes que elas sejam criadas.

O AVS (Automatic Versatile Search tool) (Santos et al., 2018) é um sistema de gerenciamento de CRs que recebe como entrada descrições de bugs encontrados pelo testador, e retorna CRs com descrições de erros semelhantes já na base indexada. A fim de tratar as variações de terminologia existentes na empresa, o sistema possui um tesauro local, que é utilizado na expansão automática de consultas. Aqui, a expansão da consulta tem dois objetivos: (1) melhorar a cobertura, recuperando todos os CRs com descrições de erro semelhantes; e (2) melhorar a classificação de CRs semelhantes, uma vez que os testadores acessam apenas os itens no topo da lista retornada.

O tesauro nativo foi construído manualmente durante o desenvolvimento do AVS. Como os usuários do sistema não têm permissão para modificar o tesauro local, com o tempo ele se tornou obsoleto, uma vez que ele não cobre novos termos oriundos das novas funcionalidades dos aparelhos móveis. Nesse cenário, constatou-se a necessidade de usar um dicionário de sinônimos externo, que pode ser recriado pelos usuários sempre que necessário. Assim, o tesauro construído neste trabalho de mestrado está sendo usado na expansão de consultas, trazendo termos novos para o contexto do AVS. Os resultados de testes preliminares, relatados na seção a seguir, foram muito positivos.

5.2.1.1 Testes comparativos

Como mencionado, realizamos testes de desempenho do AVS em cenários distintos, para avaliar a contribuição do novo tesauro para o sistema. Comparamos o desempenho do AVS em três cenários: usando apenas seu tesauro nativo, usando apenas o tesauro criado pelo Motosaurus, e usando os dois tesauros combinados.

O corpus usado nesses testes conta com aproximadamente 18 mil CRs de diferentes componentes do celular. Cada CR contava com os seguintes campos:

resumo, descrição, id, status entre outros. Além disso, as CRs continham ao menos 1 arquivo (e.g., vídeo ou imagem) para fornecer aos participantes envolvidos no estudo detalhes sobre o erro reportado.

Com o intuito de realizar esse estudo de forma controlada, simulando um ambiente de produção, contamos com o auxílio de três testadores que trabalham na empresa. Os participantes já tinham conhecimento básico sobre as funcionalidades do AVS, não sendo necessário um treinamento para seu uso. Esses testadores então criaram consultas a partir de erros detectados durante suas atividades de teste manual, submetendo-as ao AVS.

Para acessar o comportamento do sistema, cada testador criou uma descrição de erro (que originaram Q1, Q2 e Q3, respectivamente) foram submetidas a cada cenário de teste. A Tabela 1 mostra a quantidade de CRs relevantes recuperadas por cada consulta até o nível 20, e sua posição no ranking em cada cenário. Esses resultados revelaram melhorias consistentes com o uso do tesauro criado neste trabalho, usado sozinho ou combinado com o tesauro nativo.

Tabela 1: Desempenho comparativo do sistema AVS

	Tesauro nativo	**Tesauro nativo + tesauro criado**	**Tesauro criado**
Q1 # CRs relevante	6	**12**	**12**
Ranking das CRs relevante	2, 3, 4, 7, 8, 12	2, 3, 4, 5, 6, 7, 8, 9,10, 12, 13, 16	1, 2, 5, 6, 7, 9, 11, 12, 14, 15, 17, 18
Q2 # CRs relevante	4	**5**	**7**
Ranking das CRs relevante	1, 5, 6, 8	1, 5, 7, 9, 10	1, 4, 6, 8, 11, 13, 18
Q3 # CRs relevante	2	**7**	**5**
Ranking das CRs relevante	1, 6	1, 5, 6, 8, 17, 19, 20	1, 6, 9, 10, 16

Fonte: autor (2021)

É importante mencionar aqui que os usuários do AVS costumam verificar, no máximo, as 10 primeiras CRs retornadas pelo sistema. Esse comportamento pode estar relacionado ao longo tempo gasto para analisar cada CRs. Diante desse cenário, observamos a importância de buscas que consigam ordenar CRs relevantes no topo do ranking. Essa também foi a razão para nós termos analisado apenas as 20 CRs melhor ranqueadas.

A seguir, a Tabela 2 mostra as médias das métricas de Precisão e Cobertura considerando as 3 consultas na Tabela 1. Calculamos essas métricas em relação ao nível 20 na lista ranqueada retornada, pelas razões já apresentadas acima.

Tabela 2: Valores médios de métricas em relação às 20 CRs melhor ranqueadas

	Tesauro nativo	**Tesauro nativo + tesauro criado**	**Tesauro criado**
Precisão	20%	**40%**	**40%**
Cobertura	40%	**79%**	**79%**

Fonte: autor (2021)

De acordo com os dados obtidos, foi possível observar que houve melhorias no mecanismo de busca do AVS quando inserimos o tesauro criado. As consultas expandidas com novos termos trouxeram bons resultados, encontrando relatórios em posições bem próximas ao topo do ranking. Com esses resultados, podemos afirmar que nosso processo de construção de tesauro demonstra ganhos de produtividade em sistemas como o AVS.

5.2.2 O Sistema ATP

O Plano de Teste (PT) é um artefato de importância central no processo de teste de software, uma vez que descreve todos os elementos necessários para as atividades de teste que serão executadas durante todo o ciclo de vida do produto. Nesta etapa, são definidos os propósitos dos testes, a abordagem e as técnicas de teste utilizadas, o cenário de teste, os Casos de Teste (CT) a serem executados, entre outros detalhes.

Em ambientes de produção, a seleção de CTs para compor o PT geralmente é feita com base em documentos referentes ao SW que será testado. Os CTs contidos no PT devem cobrir ao máximo as funcionalidades e os erros reportados nas CRs atuais do software. Contudo, essa tarefa é muito árdua, uma vez que a quantidade de testes disponíveis para diferentes linhas de produtos é muito grande, dificultando a escolha dos melhores CTs. Além disso, geralmente esse processo é realizado manualmente por um engenheiro de teste, que muitas vezes não possui treinamento para realizar tal tarefa.

A fim de minorar esse problema, empresas de desenvolvimento de software estão buscando soluções para automatizar totalmente ou parcialmente o processo manual de criação desses planos de teste. No âmbito da Motorola Mobility, existem iniciativas que visam automatizar o processo de criação de PTs, com destaque para o sistema Auto Test Plan - ATP (Magalhães, 2020). ATP visa automatizar a criação de PTs no contexto de testes de regressão, cujo objetivo principal é verificar se as mudanças realizadas no código eliminaram os erros detectados, e se não causaram o aparecimento de novos erros.

O sistema ATP recebe como entrada Notas de lançamento (do inglês, *release notes* – RN) e CRs associadas ao produto sendo testado, e retorna CTs relacionados aos bugs descritos nesses documentos. O ATP utiliza palavras-chave automaticamente selecionadas a partir dos documentos de entrada para construir consultas usadas para recuperar CTs a partir de uma base indexada. Os CTs priorizados pela ATP irão compor o PT a ser executado manualmente. Contudo, o ATP não permite que usuários modifiquem manualmente as consultas automaticamente criadas, a fim de adicionar termos equivalentes aos que foram selecionados. Aqui também é possível identificar problemas já mencionados sobre as variações de terminologia existentes dentro do domínio.

No contexto de testes de regressão, quando CTs relevantes estão ausentes em um PT de regressão, parte do novo código não será testada. Assim, a expansão da consulta deve objetivar aumentar a *cobertura de código* dos testes selecionados, recuperando todos os CTs relevantes existentes. Note que, embora novos termos sejam adicionados à consulta, as taxas de precisão ainda podem ser mantidas altas, uma vez que o tesauro usado é específico para o domínio da aplicação.

A fim de minorar esse problema, o tesauro construído neste trabalho foi acoplado à ATP, expandindo a consulta original com termos relacionados. Os resultados obtidos, apresentados na seção a seguir, foram muito positivos.

5.2.2.1 Testes comparativos

Testes comparativos foram realizados com o sistema ATP em dois cenários diferentes: sem e com expansão de consulta com base no tesauro criado. O objetivo principal foi verificar as melhorias trazidas pela expansão das consultas na cobertura

do código alcançada pelos CTs selecionados. Para isso, usamos outro sistema local (privado) que mede a cobertura de código de CTs em um Plano de Teste.

O sistema Auto Test Coverage (ATC) acompanha os métodos que foram modificados no código-fonte, oferecendo uma maneira de monitorar os dispositivos enquanto eles estão sendo testados (Magalhães et al., 2020). Dessa forma, é possível obter o percentual de cobertura de código de um Plano de Teste para um determinado software.

Para comparar de forma justa as duas configurações do ATP (com e sem o uso do tesauro), foram executadas duas etapas de verificação de seu desempenho. Na primeira verificação, o sistema ATP recebeu como entrada 48 CRs obtidas de 10 RNs relacionadas à mesma versão do dispositivo em teste, criando automaticamente 2 planos de teste (um usando expansão de consulta e o outro não). Na segunda verificação, o ATP recebeu 28 CRs de entrada oriundas de 4 RNs, também relacionada a mesma versão do dispositivo em teste, criando 2 planos de teste como no caso anterior.

Os planos gerados foram executados manualmente por um testador, que fez o monitoramento com o sistema ATC, a fim de medir a cobertura de código de cada PT individualmente. Além disso, os CTs selecionados por cada PT foram manualmente avaliados, a fim de possibilitar o cálculo da medida de precisão do sistema. As Tabelas 3 e 4 mostram os resultados dos testes comparativos.

Tabela 3: Desempenho comparativo do ATP com 10 RNs

	Tesauro off	**Tesauro on**
# TCs recuperados	438	474
# TCs relevantes	302	334
Precisão	69%	70%
Cobertura de código	35,50%	49,29%
# TCs que falharam	20	28
# TCs que falharam entre os 100 primeiros do ranking	5	9

Fonte: autor (2021)

Tabela 4: Desempenho comparativo do ATP com 4 RNs

	Tesauro off	**Tesauro on**
# TCs recuperados	435	451
# TCs relevantes	296	302
Precisão	36,40%	52,10%
Cobertura de código	35,80%	48,27%
# TCs que falharam	13	16
# TCs que falharam entre os 100 primeiros do ranking	3	6

Fonte: autor (2021)

De acordo com os dados obtidos, foi possível observar que a expansão da consulta apresentou melhores resultados, tendo recuperado uma maior quantidade de CTs relevantes (melhorando assim as taxas de *cobertura de código*). Apesar desse aumento na quantidade de CTs recuperados, a taxa de *precisão* não foi afetada negativamente. É importante observar também que a quantidade de bugs encontrados pelo ATP com expansão de consulta foi ampliada. Não foi possível medir as taxas de *recall* nestes testes porque não conseguimos avaliar o número total de CTs relevantes na base da Empresa, que conta com milhares de CTs. Portanto, contamos apenas com a cobertura de código.

Por fim, é importante destacar que, em ambientes de produção, nem sempre é possível executar todos os CTs nos PTs gerados pela ATP. Por falta de tempo, geralmente os engenheiros de teste responsáveis por essa atividade adotam um ponto de corte, selecionando apenas os CTs que ficam no topo do ranking. Nesse cenário, é de extrema importância que os casos de testes que melhor atendam os erros reportados estejam bem posicionados no ranking criado pelo sistema. Como pode ser visto nas Tabelas 3 e 4, avaliamos a posição dos testes que falharam durante a execução do PT para os 100 primeiros CTs ranqueados pela ATP. Como pode ser observado, houve um número maior de testes que falharam nos PTs criado pelo o ATP com expansão de consulta.

5.3 CONSIDERAÇÕES FINAIS

Este capítulo apresentou detalhes do Motosaurus, o protótipo desenvolvido a partir do processo definido no Capítulo 4, para auxiliar na construção de tesauros de domínio específico. Os testes comparativos realizados confirmaram nossa hipótese inicial de que o uso de um tesauro de domínio específico é capaz de melhorar os resultados de sistemas de RI através de operações de expansão de consultas.

O capítulo 6, a seguir, traz conclusões mais gerais sobre este trabalho, bem como indicações de trabalhos futuros.

6 CONCLUSÃO

Este trabalho de mestrado propõe um processo para a geração de tesauros de domínio específico, criados a partir de documentos disponíveis nos repositórios das empresas. Os termos extraídos dos documentos de entrada são agregados em conjuntos de termos sinônimos ou relacionados. O protótipo implementado utiliza técnicas de PNL, como POS-tagging, RegEx e n-gramas, para enriquecer o dicionário de sinônimos com termos compostos, siglas e abreviações. Finalmente, os conjuntos de saída de palavras/termos relacionados são indexados e disponibilizados para consulta manual ou automática.

Nosso estudo de caso se concentrou no domínio de teste de SW. O tesauro construído foi usado na expansão automática de consultas de dois sistemas de RI da Motorola Mobility. Testes comparativos revelaram que o uso do tesauro melhorou a cobertura de recuperação e a classificação de documentos relevantes nesses dois sistemas. Esses resultados são muito importantes em cenários do mundo real, nos quais os testadores nem sempre têm tempo para analisar todos os documentos existentes.

Excetuando-se a etapa de Validação do tesauro, todas as outras etapas de processamento são automáticas. Isso confere ao protótipo possibilidades reais de uso dentro de empresas que demandem tesauros específicos, sem maiores dificuldades. Esse ponto negativo será tratado na Seção 6.2, como indicação de trabalhos futuros.

6.1 PRINCIPAIS CONTRIBUIÇÕES

Acreditamos que este trabalho traz boas contribuições tanto para a área de construção de tesauros, como para a área de Teste de Software.

- Não encontramos na literatura relaciona outro processo similar ao proposto aqui, que é capaz de não apenas gerar grupos de termos relacionados a partir de documentos de entrada, como também de enriquecê-los com termos compostos, siglas e abreviações;

- O trabalho atendeu a uma demanda real de empresas que trabalham com grandes bases de documentos de domínio específico, que não conseguem bons resultados com o uso de tesauros genéricos;
- O processo proposto é simples de entender e de implementar, mesmo que seja para domínios de aplicação diferentes de teste de SW (pois é um processo genérico);
- O protótipo implementado como prova de conceito se mostrou eficaz para a criação de um tesauro de domínio específico a partir de documentos comuns em empresas de teste de SW;
- O protótipo pode ser modificado de forma simples para aceitar como entrada documentos de outro tipo (basta modificar o processo inicial de criação do corpus);
- O tesauro criado melhorou os resultados de dois sistemas de RI através de uma técnica simples de expansão de consulta.

6.2 Trabalhos Futuros

Listamos aqui alguns trabalhos que podem ser desenvolvidos a fim de corrigir alguns pontos fracos deste trabalho, bem como de estender esse trabalho para outras aplicações.

Atualmente, a etapa de validação do tesauro é executada manualmente, sendo considerada como um ponto que necessita de melhorias no processo de geração. Assim, propomos aqui dois procedimentos simples para iniciar sua automação completa:

- Validar a ocorrência dos termos gerados no Processo II (acrônimos, siglas e abreviações) diretamente a partir do corpus de entrada. Termos que não ocorrerem em nenhum documento de entrada serão automaticamente descartados;
- Caso já exista algum tesauro anterior na empresa, os novos tesauros podem ser contrastados com os pré-existentes, e os termos verificados no tesauro existente serão automaticamente validados.

Existem ainda algumas ideias a considerar como trabalhos futuros.

- Adaptar o processo inicial para tratar outros documentos relevantes na Empresa, tais como Notas de Lançamento/Versão (*Release Notes*). Esses documentos também trazem informações relevantes para a criação dos planos de teste, como já mencionado.
- Criar uma interface para edição do Mapa Mental, de modo a facilitar ainda mais sua manutenção. Essa interface seria muito útil para a criação de Mapas Mentais para outros domínios de aplicação.
- Utilizar os links do Mapa Mental para enriquecer os conjuntos do tesauro com termos representando relações de especialização ou de generalização. Como mencionado na Seção 4.3.1, é possível criar um subconjunto de termos relacionados a partir de um nó de mais alto nível. Por exemplo, a funcionalidade camera pode dar origem ao seguinte subconjunto de termos: Camera [slow motion, zoom, flash, vídeo, photo, focusses]. Esse subconjunto pode ser usado na expansão de consultas, se desejado.
- Atualizar o tesauro criado a partir de documentos mais recentes e repetir os testes comparativos, desta vez com uma quantidade maior de consultas submetidas aos sistemas ATP e AVS.
- Utilizar o tesauro criado acoplado a outros sistemas de processamento de texto da empresa, como o classificador de casos de teste automatizáveis, e verificar as melhorias de desempenho alcançadas.
- Utilizar o protótipo para criar novos tesauros com foco na área de desenvolvimento de software da Motorola Mobility Brasil a partir de documentos de requisitos e outras informações disponíveis sobre o software.

REFERÊNCIAS

ABREU, V. S. **O Léxico na Internet: Análise de Neologismos em Comunidades do Orkut.** . In: 3 o SIMPÓSIO HIPERTEXTO E TECNOLOGIAS NA EDUCAÇÃO REDES SOCIAIS E APRENDIZAGEM. UFPE - Núcleo de Estudos de Hipertexto e Tecnologias na Educação - Recife: 2010. Disponível em: <http://nehte.com.br/simposio/anais/Anais-Hipertexto-2010/Verena-Santos-Abreu.pdf >. Acesso em: 22 jun. 2021.

ALI. N. H. and IBRAHIM. N. S. **Porter stemming algorithm for semantic checking.** in Proc. 16th Int. Conf. Comput. Inf. Technol., Jun. 2012.

ALLAHYARI, M., POURIYEH, S., ASSEFI, M., SAFAEI, S., TRIPPE, E. D., GUTIERREZ, J. B., KOCHUT, K. **A Brief Survey of Text Mining: Classification, Clustering and Extraction**. Techniques.In Proceedings of KDD Bigdas, Halifax, Canada, V.2,2017.

APPELT, D. E., ISRAEL, D. J. **Introduction to Information Extraction Technology: a Tutorial.** In the proceeding of: 16th International Joint Conference on Artificial Intelligence (IJCAI'99)16th International Joint Conference on Artificial Intelligence, Stockholm, Sweden. 1999.

ARANHA C. E PASSOS, E. **A tecnologia de mineração de textos.** Revista Eletrônica de Sistemas de Informação, ISSN 1677-3071,v. 5, n. 2, 2006.

ARANHA C., V. M. **Uma abordagem de pré-processamento automático para mineração de texto em português:** enfoque em inteligência computacional. 2007.

ALTOUNIAN, M. ; GOMES, B. **A recuperação semântica da informação no contexto do controle externo.** ***Revista do Tribunal de Contas da União***, n. 137, 2016. Disp. em: https://revista.tcu.gov.br/ojs/index.php/RTCU/article/view/1376 Acesso em: 7 jan. 2022.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Recuperação de Informação: Conceitos e Tecnologia das Máquinas de Busca**. 2. ed. [s.l.] Bookman Editora, 2013.

BLUMETTI, B.; LEMOS, C.; CARLOS, L.; ARAúJO, R. **Seleção de informações usando text mining com RI**. p. 13, 2007.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. **The semantic web.** ***Scientific American*****,** n. 1, p. 34-43, 2001.

BRITO, E. M. N. **Mineração de Textos: Detecção automática de sentimentos em comentários nas mídias sociais**. p. 54, 2016.

BONFIM, M. E. **Recuperação de Documentos -Texto Usando Modelos Probabilísticos Estendidos.** Iniciação Científica Cesumar, v. 11, n. 2, 2009.

BUGZILLA. (acessado em Julho 12, 2021), http://www.bugzilla.org/.

CAFÉ, Lígia. **Organização da documentação e da informação II: classificação e indexação.** Florianópolis: CIN/ CED/UFSC. 2010.
CAMILO, C. O. **Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas.** 2009. Disponível em: <http://www.portal.inf.ufg.br/sites/default/files/uploads/relatorios-tecnicos/RT-INF_001-09.pdf>. Acesso em: 29 set. 2021.

CAMPOS, M.L.A., GOMES, H.E. **Metodologia de elaboração de tesauro conceitual: a categorização como princípio norteador.** Perspectivas em ciência da informação, vol. 11-3, 2006.

CAVALCANTI, Y. C. **A Bug Report Analysis and Search Tool: Improving search and analysis of duplicate bug reports**. jan. 2009.

CHATTERJEE, S., GOYAL, D., PRAKASH, A., SHARMA, J. **Exploring healthcare/health-product ecommerce satisfaction: A text mining and machine learning application**. Journal of Business Research, p.815–825,12 oct. 2020.

COLLOBERT R., WESTON J., BOTTOU L., KARLEN M., KAVUKCUOGLU K. AND KUKSA P.. **Natural Language Processing (Almost) from Scratch**. Journal of Machine Learning Research, 12:2493- 2537, 2011.

CORREIA, M. F. DE B. **Recuperação de Informação.** Universidade Virtual Africana: 04 out. 2020. Disponível em: <https://oer.avu.org/handle/123456789/653>. Acesso em: 17 maio. 2018.

CORRÊA, Geraldo Nunes et al. **Uso da mineração de textos na análise exploratória de artigos científicos**. 2012. Disponível em:< http://www.icmc.usp.br/CMS/Arquivos/arquivos_enviados/BIBLIOTECA_113_RT_383.pdf>. Acesso em: 01 jul. 2020.

COWIE, J. & W. LEHNERT. **Information extraction.** Communications of the ACM 39(1), 80–91. 1996.

CHURCH, K.W.; HANKS, P.: **Word association norms, mutual information, and lexicography.** Computational Linguistics, 1990.

DAHLBERG, I. **Teoria do conceito.** Tradução de Astério Tavares Campos. International Classification. Rio de Janeiro, v. 7, n. 2, 1978.

DE ÁVILA, R. L. F.; SOARES, J. M. **Uso de técnicas de pré-processamento textual e algoritmos de comparação como suporte à correção de questões dissertativas: experimentos, análises e contribuições**. : Simpósio brasileiro de informática na educação, nov. 2013. Disponível em: <http://www.br-ie.org/pub/index.php/sbie/article/view/2551>. Acesso em: 04 out. 2020.

DICE, LEE R. **Measures of the Amount of Ecologic Association between Species**. Journal of Ecology, 1945.

DODEBEI. V.: **Tesauro: linguagem de representação da memória documentária**, Niterói: Intertexto, 2002.

DOS SANTOS, J. C.; SIQUEIRA, S. **Extração de Informações Baseada em Ontologia: Oportunidades e Desafios.** RelaTe-DIA, *[S. l.]*, v. 12, n. 1, 2019. Disponível em: http://www.seer.unirio.br/monografiasppgi/article/view/9106. Acesso em: 20 jan. 2022

DUARTE, Elizabeth Andrade. **Processos de indexação e análise de assunto: uma abordagem baseada na avaliação dos fatores intervenientes nestes processos**. Biblionline, João Pessoa, v.3, n.2, jul./dez 2007.

EGAN. J.P. **Signal detection theory and ROC analysis**. New York, USA: Academy Press, 1975

FAYYAD, U; PIATETSKY-SHAPIRO, G; SMYTH, P. **From data mining to knowledge discovery in databases.** Ai Magazine, 17:37–54, 1996.

FONSECA, E. R., ROSA, J. L. G. AND ALUÍSIO, S. M. **Evaluating word embeddings and a revised corpus for part-of-speech tagging in portuguese.** Journal of the Brazilian Computer Society, 2015.

FOSKETT, D. **Thesaurus.** In: Subject and information analysis. New York: M. Dekker, 1985.

JURASKA J., KARAGIANNIS P., BOWDEN K., AND WALKER A. 2018. **A deep ensemble model with slot alignment for sequence-to-sequence natural language generation**. arXiv preprint arXiv:1805.06553.

JUNG, H., LEE, B. G. **Research trends in text mining: Semantic network and main path analysis of selected journals**. ELSEVIER, South Korea, 2020.

GABRIEL JUNIOR,R. F.; LAIPELT, R. C. F. (2017). **Thesa: ferramenta para construção de tesauro semântico aplicado interoperável**. Revista P2p e Inovação. 3:2 (2017) 124-145.

GRANADA. R. L.: **Processos de construção automática de tesauro**, Porto Alegre, 2011.

GREFENSTETTE, G.: **Explorations in Automatic Thesaurus Discovery**. Kluwer Academic Publishers, 1994.

GUPTA, Vishal; LEHAL, Guruprit. **A survey of text mining techniques and applications.** In: . [S.l.: s.n.],v.1, 2009.

GOMES, R. M. **Desambiguação de Sentido de Palavras Dirigida por Técnicas de Agrupamento sob o Enfoque da Mineração de Textos.** Pontifícia Universidade Católica do Rio de Janeiro. Rio de Janeiro, 2009.

Glickman, Mark E. and Jones, Albyn C.: **Rating the Chess Rating System**, Chance 12, 1999.

GONZALEZ, M.; LIMA, V. L. S. **Recuperação de Informação e Processamento da Linguagem Natural.** In: XXIII CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 2003, Campinas, 2003.

GRAINGER, T. and POTTER, T. **Solr in Action**. Manning Publications Co., 2014.

HOTHO, A.; NüRNBERGER, A.; PAASS, G. **A brief survey of text mining.** LDV Forum - GLDV Journal for Computational Linguistics and Language Technology, v. 20, 2005.

INDURKHYA, N. and DAMERAU, F. J. **Handbook of Natural Language Processing.** Chapman & Hall/CRC, 2nd edition. 2010.

JACKSON, P.; MOULINIER, I. **Natural language processing for online applications: Text retrieval, extraction and categorization.** [S.l.]: John Benjamins Publishing, 2007.

JAIN, A. K., DUBES, R. C. **Algorithms for Clustering Dat**a. Editora Prentice-Hall. Jolliffe, I. T. (2002). Principal Component Analysis. Springer, Segunda edição, 1988.

JONES, D. B.; SOMERS, H. **New Methods In Language Processing.** [S.l.]: Routledge, 2013.

KAJI, H.; MORIMOTO, Y., AIZONO, T.; Yamasaky, N. **Corpus dependent association thesauri for information retrieval**, In: Proceedings of the 18th Conference on Computational Linguistics, 2000.

KRISHNAN V. AND MANNING C. **An Effective Two-Stage Model for Exploiting Non-Local Dependencies in Named Entity Recognition**. In ACL, 2006.

KAUR, P., MADAN, N. **Technique of Text Mining: A Survey.** International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS), p.513–519, 2015.

KADHIM, A. I. **Survey on supervised machine learning techniques for automatic text classification**. Artificial Intelligence Review, v. 52, n. 1, p. 273–292, 2019.

LOPES, Maria Célia Santos. **Mineração de dados textuais utilizando técnicas de clustering para o idioma Português.** Rio de Janeiro, 2004.

LESKOVEC, A.; RAJARAMAN, A.; ULLMAN, J. D. **Mining of Massive Datasets.** Cambridge: Cambridge University Press, 2014.

LIU, B. and ZHANG, L. **A survey of opinion mining and sentiment analysis,** pages 415–463. Springer US, 2013.

LISBACH, B. AND MEYER, V. **Linguistic identity matching**. Springer, 2013.

MADUREIRA, F. A. P. **Classificação de Documentos.** THESIS.MASTER—Lisboa: Universidade Nova de Lisboa, 2009.
MAGALHÃES C, ANDRADE J, PERRUSI L, MOTA A, BARROS F, AND MAIA E. 2020. HSP: **A hybrid selection and prioritisation of regression test cases based on information retrieval and code coverage applied on an industrial case study.** Journal of Systems and Software, 2020.

MACIEIRA, L. G. **Metodologia ágil para gestão e planejamento de projetos: implantação do Scrum e do framework Jira na qualidade em uma empresa do setor aeronáutico**. In: SIMPÓSIO DE ENGENHARIA DE PRODUÇÃO DE SERGIPE, 10., 2018, São Cristóvão, SE. Anais [...]. São Cristóvão, SE, 2018.

MARTINS, C. A.; GODOY, D.; MONARD, M. C.; MATSUBARA, E. T.; AMANDI, A. **Uma experiência em mineração de textos utilizando clustering probabilístico clustering hierárquico**, 2003.

MATOS E. **DRL TC Classifier: Classificação de casos de teste para automação de testes de dispositivos móveis,** Dissertação (Mestrado). Universidade Federal de Pernambuco, Recife*,* 2020.

MATOS, P., **Relatório técnico "Métricas de avaliação”**, Tech. rep.,Universidade Federal de São Carlos (UFSCar), sep 2009.

MEDEIROS, D. M. R. **Análise de dados por meio de agrupamento fuzzy semisupervisionado e mineração de textos**. 2010. 111 p. TESE (Doutorado em Ciências - Ciências de Computação e Matemática Computacional) - Instituto de Ciências Matemática e de Computação - ICMC - USP, São Carlos, SP, 2010.
MILLER, G.: WordNet: **A lexical database for the english language**. On Communications of the ACM. 1995.

MIRANDA, L. M. C.; MEDEIROS, M. B.; SUJII, M. K. **Elaboração de tesauros utilizando-se o programa de elaboração de tesauros em microcomputador (Tecer)**. Bibliotecon. Brasília, v. 18, n. 2, 1990. Disponível em: <http://164.41.105.3/porta.../RBB/article/view/649/646>. Acesso em: 29 maio 2010.

MYERS, G. J., SANDLER, C., & BADGETT, T. **The art of software testing**. John Wiley & Sons, 2011.

MONARD, M. C.; BARANAUSKAS, J. A. **Conceitos sobre aprendizado de máquina. Sistemas Inteligentes-Fundamentos e Aplicações**, v. 1, n. 1, 2003.

NGUYEN, A. T. et al. **Duplicate bug report detection with a combination of information retrieval and topic modeling.** International Conference on Automated Software Engineering, 2012. Anais. New York, USA, 2012. Disponível em: <http://dl.acm.org/citation.cfm?doid=2351676.2351687>.
Acesso em: 16 abr. 2021.

RAJANI T. **Importance of Testing in Software Development Life Cycle.** International Journal of Scientific & Engineering Research, 2012.

RAHMAN M. AND ROY C. **Effective reformulation of query for code search using crowdsourced knowledge and extra-large data analytics**. In ICSME, 2018.

ROJAS R., **Unsupervised learning and clustering algorithms**, in Neural Networks. Springer, 1996. Disponível em: https://link.springer.com/chapter/10.1007/978-3-642-61068-4_5 . Acesso: 29 maio 2021.

RUSSO, V. **Clustering and classification in Information Retrieval: from standard techniques towards the state of the art**. Technical Report TR-9-2008 – SoLCo Project, 2008.

SANTOS I., ARAÚJO J., LIMA C., PRUDÊNCIO R., AND BARROS F. 2018. AVS: An Approach to Identifying and Mitigating Duplicate Bug Reports. In Proceedings of the XIV Brazilian Symposium on Information Systems (Caxias do Sul, Brazil) (SBSI'18). ACM, NY, USA, Article 22, 7 pages. https://doi.org/10.1145/3229345.3229370

SEBASTIANI, F. **Machine learning in automated text categorization.** [S.l.]: ACM Computing Surveys. v. 34, n. 1. 2002, 47 p. Disponível em <http://nmis.isti.cnr.it/sebastiani/Publications/ACMCS02.pdf>. Acesso em 05 set. 2018.

SIMOES, S., P. DEEPAK, M. SAIRAMESH, D. KHEMANI, AND S. MEHTA. **Content and context: Two-pronged bootstrapped learning for regex-formatted entity extraction**., in AAAI, 2018.

SILVA, Eduardo Fraga do Amaral. **Um sistema para extração de informação em referências bibliográficas baseado em aprendizagem de máquina.** Dissertação (Mestrado). Universidade Federal de Pernambuco, Recife, 2004.

SOHRAWARDI, S. J.; AZAM, I.; HOSAIN, S. **A comparative study of text classification algorithms on user submitted bug reports.** Ninth International Conference on Digital Information Management, 2014. Disponível em: <http://ieeexplore.ieee.org/document/6991434/>. Acesso em: 05 out. 2020.

SOARES, F. D. A. **Mineração De Textos Na Coleta Inteligente De Dados Na Web.** Pontifícia Universidade Católica do Rio De Janeiro - PUC-RIO, 2009.

SOARES, A. M. A **Mineração de texto na analise de contas publicas municipais.** Dissertação (Mestrado Profissional em Computação aplicada) Universidade Estadual do Ceará, Fortaleza, 2010.

PASSINI, M. L. C. **Mineração de Textos para Organização de Documentos em Centrais de Atendimento**. Dissertação (Mestrado em Engenharia Civil) - Instituto Alberto Luís Coimbra de Pós-graduação e Pesquisa de Engenharia, Universidade Federal do Rio de Janeiro, UFRJ. Rio de Janeiro, 2012.

PARK. Y. C., HAN. Y. S., AND CHOI. K.-S. **Automatic thesaurus construction using bayesian networks**. Information Processing and Management, 1996.

PEREIRA, E. C. ; BUFREM, L. S. **Princípios de organização e representação de conceitos em linguagens documentárias**. Encontros bibli, Florianópolis, n. 20, 2. semestre, 2005.

PEZZINI, Anderson. **Mineração de Textos: Conceito, Processo e Aplicações**. Revista Eletrônica do Alto Vale do Itajaí, Santa Catarina, v. 5, n. 8, p.1-13, dez. 2016.

PRIHATINI. P. M., PUTRA I. K. G. D., GIRIANTARI I. A. D., and Sudarma M., **"Stemming Algorithm for Indonesian Digital News Text Processing,"** Int. J. Eng. Emerg. Technol., vol. 2, no. 2, pp. 1–7, 2017.

VAN DER LAAN, R. H. **Tesauro e terminologia: uma inter-relação lógica**. 2002. Tese (Doutorado em Letras) – Universidade Federal do Rio Grande do Sul, Porto Alegre, 2002.

Xu R, Wunsch DC. **Clustering**. IEEE Computer Society Press; 2008.

WAZLAWICK, R. **Engenharia de Software: Conceitos e Práticas.** Elsevier/Campus, 2013.

WILLET, P. **The Porter stemming algorithm: then and now**. Electronic Library and Information Systems, The University of Sheffield. Sheffield, 2006.

ZACARIAS, R. O., JACYNTHO, M. D. D. A., & VASCONCELOS, A. P. V. D. (2020). **Um método para criação de tesauros semânticos de requisitos usando a ontologia SKOS**. *Transinformação*, *32*.