Processamento de Linguagem Natural para Mapeamento da Estrutura Curricular de Formação em Engenharia Mecatrônica na UnB à Legislação Vigente

Wallace Ben Teng Lin Wu
Departamento de Ciência da Computação
Universidade de Brasília
Brasília, Brasil
wallacebtlw@gmail.com

***Resumo*—A legislação vigente no Brasil que regulamenta o trabalho feito por profissionais no ramo da Engenharia é extensa e complicada. Por meio da atribuição de títulos, o sistema CREA/CONFEA define quais competências cada engenheiro possui e, portanto, quais são as suas responsabilidades e os conhecimentos técnicos exigidos na sua formação. Desse modo, é de interesse das instituições de ensino superior do país e também dos alunos em engenharia, compreender quais os conteúdos técnicos que devem ser ministrados durante a graduação e como as disciplinas ofertadas suprem essa necessidade. Atualmente, não há um artefato capaz de fazer esse relacionamento rapidamente e com a versatilidade para ser utilizado por vários cursos e universidades. Sabendo disso, esse projeto propõe a elaboração de um programa que faça, de modo eficiente e acurado, o mapeamento das disciplinas ofertadas por um curso qualquer com os conteúdos técnicos exigidos pela legislação. Com essa ferramenta desenvolvida, tornou-se possível facilmente fazer buscas por quais os tópicos que estão sendo cobertos por uma matéria ou o processo inverso. Além disso, ela foi construída com técnicas de processamento de linguagem natural de tal modo que a torna versátil e flexível, podendo ser expandida para diversos cursos e universidades também.**

***Palavras-Chaves*—Processamento de Linguagem Natural, CREA, Estrutura Curricular, Similaridade Semântica, Latent Semantic Indexing.**

I. Introdução

De acordo com a legislação vigente, a formação de engenheiros no Brasil deve seguir uma série de normas regulamentadas pelo Conselho Federal de Engenharia e Agronomia (CONFEA) e pelo o Ministério da Educação (MEC), sendo elas reforçadas pelos Conselhos Regionais de Engenharia e Agronomia (CREA) de cada Estado. Essa regulamentação é bem complexa e abrangente, definindo desde as carga horárias mínimas [1] de graduação dos cursos até o perfil de egresso e os conhecimentos que devem ser ministrados pelos cursos [2]. A existência dela é indispensável, pois estabelece um padrão de qualidade para os cursos de Engenharia, garantindo que os profissionais formados no Brasil estão aptos para atuar no mercado de trabalho. Os engenheiros são profissionais com muitas responsabilidades, possuindo a capacidade de impactar profundamente na sociedade, seja positivamente, como desenvolver métodos eficientes de geração de energia limpa para a população, ou negativamente, como provocar danos ambientais que podem perdurar por diversas gerações.

No Brasil, há o total de 22 cursos de engenharia regulamentadas pelo CONFEA e o MEC [2], cada um com suas respectivas características individuais, abrangendo diferentes áreas de conhecimento, atuando em diferentes campos do trabalho e possuindo distintas responsabilidades. Essa legislação é dividida e detalhada em diversos documentos, como os *Referenciais Nacionais dos cursos de Engenharia*, os quais definem como cada curso deve ser abordado pelas universidades [2]; as *Diretrizes Curriculares Nacionais do Curso de Graduação em Engenharia*, que listam os direitos e os deveres de cada profissional [3]; a *Resolução N° 1.073, de 19 de Abril de 2016*, que regulamenta a atribuição de títulos, atividades, competências e campos de atuação profissionais [4]; e entre vários outros [1], [5]. O modelo adotado pelo governo separa o mercado de trabalho em diversos campos, ou áreas de atuação, exigindo que, para ser capaz de trabalhar em certa área, o engenheiro deve ser formado em uma instituição de ensino superior regulamentada e também possuir as competências para trabalhar nesse setor [4].

Consequentemente, toda competência exige um determinado conjunto de conhecimentos técnicos característicos para poder ser incluído no título profissional do engenheiro formado, variando desde temas como *Noções de Direto* até *Cálculo Diferencial e Integral*. Assim, nota-se a extensão da quantidade de conteúdos técnicos que existem e são requisitados para cada tipo de competência. Além disso, cada instituição de ensino no Brasil possui autonomia para definir seu projeto pedagógico de curso, desde que aderente à legislação vigente. Por conseguinte, os diversos cursos de engenharia ofertados no Brasil acabam sendo construídos de modos distintos, sendo compostos por diversas e diferentes disciplinas, mesmo abordando os mesmos conhecimentos exigidos pela

Esse trabalho foi realizado com o apoio financeiro do Conselho Nacional de Desenvolvimento Científico e Tecnológico.

regulamentação. Isso pode ser visto, por exemplo, comparando a estrutura curricular do curso de Engenharia Mecatrônica da UnB [1] e da USP [2], que diferem demasiadamente entre si, nas próprias disciplinas e nas ementas. Sabendo disso, percebe-se que há uma dificuldade presente nesse sistema atual da legislação: ter conhecimento de quais são as disciplinas de um curso que suprem os conteúdos exigidos para cada competência.

Qualquer estudante de engenharia que deseje saber quais os conhecimentos necessários para estudar e poder trabalhar em uma certa área, por exemplo, poderia gastar diversas horas pesquisando sobre a legislação vigente e como ela se relaciona com as ementas das disciplinas de sua universidade. Além disso, também é de interesse das universidades sempre melhorar o seu ensino. Assim, conhecendo-se os conteúdos técnicos exigidos para cada competência e título profissional do CONFEA, as instituições de ensino superior se tornam mais aptos a adaptar os projetos pedagógicos dos cursos ofertados. Consequentemente, pode-se elaborar mudanças nas estruturas curriculares dos cursos, descartando as disciplinas desnecessárias, acrescentando outras essenciais e modificando as ementas para que seja garantida a qualidade de ensino aos alunos e que eles sejam competentes ao se formarem.

A Universidade de Brasília (UnB) também visa sempre estar de acordo com a legislação vigente e melhorar o ensino oferecido. Pensando nisso, esse projeto, inicialmente desenvolvido com o fluxo de matérias do currículo do curso de Engenharia Mecatrônica da UnB, utiliza-se do processamento de linguagem natural para construir um programa que mapeia os conteúdos técnicos exigidos pela regulamentação vigente às matérias disponíveis na UnB. Com esse programa, torna-se bem mais prático e rápido o trabalho de encontrar as ligações entre as ementas e os conhecimentos exigidos. Consequentemente, é facilitado a verificação da regularidade dos cursos pelos administradores da UnB e também as buscas dos alunos pelo o que é preciso estudar para estarem aptos a trabalhar no mercado de trabalho. Da mesma forma, profissionais já formados podem também buscar por cursos específicos para suprir os conhecimentos que lhes faltam e são requeridos para adquirir certas competências necessárias para expandir a sua área de atuação.

II. Revisão de Literatura

Nessa seção, serão apresentados com mais detalhes como a legislação do Brasil regulamenta os cursos de engenharia e também a literatura estudada para o desenvolvimento do programa proposto nesse projeto, que implementa a funcionalidade de mapeamento entre disciplinas e conteúdos exigidos pelo MEC.

A. Legislação vigente

Primeiramente, é importante saber que o sistema CREA/CONFEA é responsável pela elaboração da regulamentação e também da fiscalização dos trabalhos relacionados à engenharia no Brasil. Essa autarquia é essencial para garantir a qualidade dos trabalhos sendo realizados, os quais podem afetar profundamente a sociedade. Sabendo disso, todos os projeto de engenharia, como as obras de construção civil, devem ser autorizados, estudados e assinados por um engenheiro responsável daquele setor. Desse modo, percebe-se a importância das responsabilidades encarregadas aos engenheiros e como é fundamental a correta formação desses profissionais.

Nos *Referenciais Nacionais dos Cursos de Engenharia* [2], é detalhado o funcionamento de cada curso de engenharia no Brasil, descrevendo o perfil de egresso esperado do profissional formado, os conhecimentos técnicos que devem ser estudados na formação, as áreas de atuação em que são capacitados para trabalhar, as competências abrangidas e também a carga horária mínima para graduação. Todas essas características são consideradas no título profissional de um engenheiro, atribuição que sintetiza as capacidades e responsabilidades de todos esses profissionais formados.

A competência é uma atribuição dada pelo CREA ao título profissional de um engenheiro que confirma a capacidade técnica e legalidade que essa pessoa possui para trabalhar naquela área [4]. Desse modo, os cursos de engenharia consistem simplesmente em um conjunto de competências relacionadas, as quais, ao serem agregadas, compõem um título profissional. Desse modo, um engenheiro de controle e automação, por exemplo, está capacitado para trabalhar com automação residencial, de fábricas ou na área de robótica, pois possui todas essas competências incluídas no seu título profissional.

As competências podem ser divididas em duas principais categorias: as que são básicas para todo engenheiro, como "ter visão holística e humanista, ser crítico, reflexivo, criativo, cooperativo e ético e com forte formação técnica" ou " estar apto a pesquisar, desenvolver, adaptar e utilizar novas tecnologias, com atuação inovadora e empreendedora" [3]; e as que são específicas para cada curso, como "O Engenheiro de Controle e Automação é habilitado para trabalhar em concessionárias de energia, automatizando os setores de geração, transmissão ou distribuição de energia" [2], competência específica ao curso de Engenharia de Controle e Automação, também chamado de Engenharia Mecatrônica. Desse modo, todo engenheiro inscrito no CREA está habilitado para trabalhar, por exemplo, como pesquisador, estudando e desenvolvendo novas tecnologias. Porém, nem todo engenheiro pode ser o responsável técnico de um trabalho envolvendo concessionárias de energia e a automação de seus setores.

Mesmo havendo essa especialização de competências nos cursos, caso algum engenheiro deseje trabalhar em uma área que foge do seu escopo de formação, ele pode solicitar a inclusão da competência responsável e uma extensão da sua área de atuação. Para isso, o profissional tem que adquirir os conhecimentos técnicos necessários relacionados à competência relativa a essa área de trabalho e, posteriormente, procurar o CREA [4] em que foi cadastrado para acrescentar

[1] https://sig.unb.br/sigaa/public/curso/ppp.jsf?lc=pt_BR&id=414333

[2] https://uspdigital.usp.br/jupiterweb/listarGradeCurricular?codcg=18&codcur=18250&codhab=0&tipo=N

a competência ao seu título profissional.

B. Processamento de Linguagem Natural

Com o objetivo de desenvolver a funcionalidade de mapeamento proposta, buscou-se tecnologias do estado da arte capazes de resolver esse problema. Dentro do escopo da inteligência artificial, o processamento de linguagem natural (PLN) é o ramo que estuda o uso de processamento computacional para resolver problemas relacionados a linguagem natural, como o português ou o inglês [6]. Abrangendo ideias da linguística computacional, estatística, álgebra linear e aprendizagem de máquina, o PLN é capaz de ler um texto escrito ou escutar um texto oral e processar esses dados para um formato que o computador consiga entender, extraindo no processo informações relevantes dos documentos [7].

Com isso, é possível gerar diversas funcionalidades, como ferramentas de tradução simultânea, de detecção de plágio ou de identificação de mensagens automaticamente geradas [8]. Para esse projeto, o problema de mapear a estrutura curricular de um curso qualquer com os conteúdos exigidos pela legislação vigente pode ser convertido em um problema de classificação semântica, em que diferentes documentos devem ser categorizados conforme os seus significados. Os documentos são unidades que contêm blocos de texto com significado completo e documentos que contemplam o mesmo conteúdo são considerados semanticamente similares. Desse modo, pode-se elaborar documentos para cada conteúdo técnico exigido pelo MEC e relacioná-los aos documentos das ementas das matérias de um curso. Caso seja identificado que uma ementa possui similaridade semântica com um conteúdo técnico, gera-se um indicativo de que a disciplina que possui essa ementa aborda aquele conhecimento técnico. Em outras palavras, é possível mapear as disciplinas com os conteúdos analisando as similaridades entre documentos.

Para o problema de classificação, existem diversos modelos e algoritmos que podem ser utilizados. Cita-se, por exemplo, o algoritmo de *k-nearest neighbours*, *Support Vector Machine*, *Decision Tree*, ou até mesmo *Deep Learning* [9]. Nesse projeto, que consiste em um problema mais específico de classificação semântica de documentos, escolheu-se o algoritmo de *Latent Semantic Analysis* (LSA), também conhecido por *Latent Semantic Indexing* (LSI). Essa decisão se baseia no fato de que o LSI é um algoritmo simples, fácil de ser implementado e que já é especializado para problemas de PLN. Ademais, estudou-se diversos projetos que já utilizaram esse algoritmo e obtiveram ótimos resultados. Alguns deles serão citados na seção de "Trabalhos Correlatos" II-D.

O funcionamento desse algoritmo se baseia na ideia de que documentos com significados semelhantes tendem a compartilhar as mesmas palavras, seja por meio da própria palavra ou de sinônimos [10]. Desse modo, objetivando identificar e agrupar as palavras relacionadas entre si, o LSI utiliza manipulações da álgebra linear para encontrar os tópicos latentes de um texto [11], que podem ser compreendidos como os diferentes temas ou conceitos contidos em um documento. Mais detalhadamente, cada um desses tópicos latentes são representados por um aglomerado de palavras relacionadas e uma atribuição de pesos para cada termo, definindo o quanto essa palavra influencia nesse tema. Assim, um documento possuir grande relevância com certo tópico latente indica que o seu texto possui muitas ocorrências de palavras abrangidas por tal tópico. Além disso, o termo "latente" indica que os tópicos não são escolhidos de antemão e sim encontrados e definidos pelo próprio algoritmo [12].

C. Ferramentas

Definido, então, o algoritmo principal que será utilizado para resolver o problema, necessita-se agora escolher as ferramentas para a implementação. Dentro da indústria e no meio acadêmico, no escopo de PLN, utiliza-se bastante as bibliotecas do Gensim [3], *Natural Language Toolkit* (NLTK) [4] e do spaCy [5], os quais possuem fortes embasamentos teóricos e já contêm muitos dos algoritmos e ferramentas considerados como o estado da arte [13], [14].

Todas essas bibliotecas são implementadas na linguagem de programação Python. Desse modo, optou-se por utilizar essa linguagem de programação nesse projeto. Além disso, essa linguagem também se destaca devido à fácil legibilidade do código, simplicidade de sintaxe e semântica [15]. Ademais, vale mencionar que o algoritmo do LSI já está implementado no Gensim e que o Python também conta com eficientes bibliotecas para manipulações matemáticas de vetores e matrizes, operações muito utilizadas no PLN, como o Numpy[6] e Pandas[7].

D. Trabalhos Correlatos

No contexto de mapeamento da legislação vigente dos cursos de engenharia às estruturas curriculares de uma universidade, já se realizaram alguns estudos acerca desse tema, dissertando-se o quão efetivo é o projeto pedagógico de um curso com os requerimentos do sistema CONFEA/CREA. Um trabalho que fez isso foi realizado no CEFET-MG em 2006 [16], o qual também abrange um curso de Engenharia Mecatrônica. Esse trabalho primeiramente sintetizou os conteúdos básicos e específicos exigidos pela legislação para o curso. Com isso, propôs-se uma divisão de todo o conhecimento exigido em 11 "Eixos", que seriam um agregado de conteúdos categorizados conforme às áreas de conhecimento que os abrangem. Assim, o projeto pedagógico do curso poderia ser mais flexível e moldaria as estruturas curriculares das disciplinas conforme a alocação disponível de docentes e a demanda dos alunos.

Há também outro artigo publicado que descreve como as competências exigidas pelo MEC se relacionam com a estrutura curricular do curso de Ciência da Computação da UFRJ [17]. Esse trabalho propôs uma metodologia iterativa de mapeamento das competências com as disciplinas ofertadas

[3]https://radimrehurek.com/gensim/index.html
[4]https://www.nltk.org/index.html
[5]https://spacy.io/
[6]https://numpy.org/
[7]https://pandas.pydata.org/

pelo departamento do Instituto de Ensino Superior da UFRJ. Descrita na metodologia desse trabalho, o processo de relacionamento foi feito de modo manual, analisando os projetos pedagógicos dos cursos e coletando os dados com os professores e alunos da universidade. Em seguida, os resultados gerados foram utilizados para validar a estrutura vigente do curso e também para reformular o projeto pedagógico para melhor abranger a legislação.

Em ambos esses trabalhos citados, o mapeamento foi feito de modo manual, desde a extração das ementas e a coleta dos dados até o relacionamento dos conteúdos técnicos das competências. Desse modo, aloca-se um tempo elevado para fazer o mapeamento e também há um esforço custoso para realizar esse mapeamento, dificultando a abrangência de outros cursos e universidades. É importante mencionar também que os projetos pedagógicos dos cursos são constantemente adaptados e modificados, exigindo um novo mapeamento toda vez que há alterações.

Outro ponto a ser considerado é que o processo manual está propenso a erros humanos, podendo prejudicar os resultados gerados. Sabendo disso, o projeto sendo discutido nesse artigo busca uma solução automatizada, que consegue facilmente ser expandida para mais cursos, mudanças de projetos pedagógicos ou outras universidades, requerendo um esforço pequeno. Ademais, esse processo automatizado reduz a influência do ser humano no processo, diminuindo os erros e restringindo o viés no mapeamento.

Desse modo, o artefato proposto por esse projeto reduz a necessidade de interferência humana e o esforço necessário para realizar o mapeamento, acelerando todo o processo e podendo ser utilizado por qualquer curso, desde que se obtenha as ementas da estrutura curricular. Essa versatilidade e agilidade só são possíveis devido ao uso do processo automatizado disponibilizado pelo PLN, que acaba fazendo esse trabalho demorado e manual de mapeamento no lugar do usuário.

Já analisando o modelo de LSI escolhido, pode-se encontrar diversos trabalhos que também obtiveram sucesso com o uso desse algoritmo. Pode-se citar dois trabalhos que propõem a elaboração de um programa que mapeia as redações escritas por alunos com um relatório chave, que serve como base para a correção das redações [18], [19]. Esse modo de correção automatizada diminuiu significativamente o tempo necessário e o esforço gasto pelos docentes. Ambos fizeram um comparativo da ferramenta proposta com a correção manual pelos professores e se observou uma alta correlação entre as notas distribuídas, enfatizando a acurácia do LSI e da sua velocidade.

Finalmente, também vale mencionar um projeto que fez a detecção de plágio entre documentos [20]. Para essa tarefa, utilizou-se o LSI para identificar a similaridade semântica entre diferentes documentos. Caso mais de um documento fosse agrupado pelos tópicos latentes e gerasse um alto valor de similaridade, eles identificariam o plágio.

Como se pode observar pelos trabalhos expostos, o LSI é muito adequado aos problemas de similaridade semântica e classificação de documentos. Ademais, essa tarefa que busca agrupar documentos similares pelo significado é justamente o problema que precisa ser resolvido nesse projeto de mapeamento da legislação com as disciplinas de um curso. Além disso, os resultados dos trabalhos citados demonstram que o algoritmo consegue efetivamente diminuir o esforço necessário para fazer o mapeamento entre documentos e, assim, reduzir o tempo gasto [18], [19]. Isso mantendo a acurácia do processo, obtendo resultados semelhantes ao trabalho manual ou até melhores.

III. Teoria e Implementação

Todo o código implementado e os documentos utilizados estão disponibilizados no Github[8]. A seguir, serão explicadas as teorias que fundamentam todos os algoritmos e modelos utilizados e o modo como implementou-os. Também dividiu-se a seção pelas etapas da implementação do programa.

A. Origem dos Dados

Primeiramente, como qualquer outro projeto de PLN, é de extrema importância ter um bom conjunto de dados [21], ou seja, dados de origens confiáveis, bem redigidos, com o mínimo de erros de digitação e contendo informações relevantes para o projeto a ser construído. Um bom modelo não é capaz de ser construído sem um conjunto de dados apropriado [22]. Consequentemente, cerca de 80% do tempo de um projeto é geralmente alocado para a aquisição e o pré-processamento dos dados [23]. Sabendo disso, adquiriu-se do CREA os documentos que relacionam os conteúdos necessários para cada competência e foi feita uma extração cautelosa das ementas das disciplinas pelo site do Sistema Integrado de Gestão de Atividades Acadêmicas (SIGAA) da Universidade de Brasília[9]. Com isso, constrói-se o corpo de documentos, que consiste no aglomerado de todos os documentos que compõem os dados [21].

B. Pré-processamento

Adquiridos os dados, prossegue-se à etapa de pré-processamento dos textos para garantir a qualidade e a relevância das informações e para moldar os dados em um formato que seja de fácil manipulação para os algoritmos. Primeiramente, os erros de digitação e as formatações incorretas das ementas foram corrigidos, etapa conhecida como limpeza dos dados, que foi realizada com a biblioteca do Gensim. A seguir, eliminou-se os *stopwords*, palavras que não agregam ou contribuem muito pouco para o significado de um documento. Portanto, essas palavras são irrelevantes para encontrar as similaridades entre documentos e a sua eliminação aumenta significativamente a acurácia do programa [13], [24]. As *stopwords* da língua portuguesa foram adquiridas do NLTK, que já contém um banco de dados dessas palavras. Além dessas, baseando-se nos resultados obtidos, adicionou-se mais termos do corpo de documentos que também deveriam ser classificados como *stopwords*. Essas palavras geralmente consistem de conectivos, preposições ou numerais, como, por exemplo, os termos “isso”, “estava” ou “havia”. Depois

[8]https://github.com/gnramos/crea-tools

[9]https://sig.unb.br/sigaa/public/

disso, converteu-se os textos para um formato padrão de fácil leitura pelo computador com o auxílio da biblioteca Gensim, deixando todas as palavras na forma minúscula e sem acentuação, além de retirar palavras de uma única letra, as quais só podem ser conectivos ou preposições na língua portuguesa e, consequentemente, não agregam significado a um texto [24], [25].

Em seguida, buscou-se uma metodologia de normalização de palavras para unir termos semelhantes e, assim, melhorar a acurácia do modelo que será treinado [26]. Isso porque o agrupamento de palavras que possuem a mesma semântica diminui a quantidade de tópicos latentes de mesmo significado que um documento contêm, unificando tópicos latentes que abrangem os mesmos conceitos.

Consequentemente, documentos diferentes que abordam o mesmo tema, mas possuem apenas a presença em comum de sinônimos ou de inflexões nas palavras, podem ser relacionados pelo LSI, pois a normalização irá considerar todas essas variações das palavras como um mesmo termo [13]. O que a normalização faz é retirar as inflexões das palavras e juntar os chamados lexemas, que são os derivados de uma determinada palavra base.

Para a normalização, considerou-se a utilização dos dois processos mais comuns: a lematização e o *stemming*. A lematização converte termos de mesma semântica e morfologia, mas com diferentes inflexões, para o lema ou raiz, o qual representa a forma base de uma família de palavras com o mesmo significado [27], [28]. Para isso, primeiramente se faz uma análise estrutural do texto e reconhece as classes morfológicas de cada palavra [29]. Em seguida, baseando-se em um dicionário detalhado que já relacionou os lemas com as suas inflexões em diversos contextos, converte-se os lexemas para a própria raiz. Por exemplo, os lexemas “gata”, “gatos”, “gatas” são convertidos para os lemas “gato” e os lexemas “tiver”, “tenho”, “tinha”, “tem” são unidos para o lema “ter”.

O outro método de normalização é o *stemming*, o qual busca reduzir as palavras para o *“stem”* delas, que é o tronco ou radical da palavra. O método do *stemming* consiste em seguir um algoritmo predefinido com uma série de regras, como o algoritmo de Porter [30], para retirar as inflexões das palavras pelo significante da palavra, ou seja, eliminam-se as letras finais de uma palavra até chegar no tronco [31]. Um exemplo dessa metodologia pode ser visto pelas inflexões “amigos” e “amigas”, as quais são convertidas para o termo “amig” pelo processo do *stemming*. Nota-se que os resultados dessa técnica podem gerar uma palavra incompleta, formando somente um agregado de letras sem possuir significado.

Devido ao processo intrínseco que cada método utiliza, a lematização é mais acurada que a *stemming*, enquanto a *stemming* é mais veloz [32], [33]. Além disso, como a lematização averígua a estrutura em que se encontra a palavra, é possível que ela diferencie palavras ambíguas que dependem do contexto. Um exemplo é o termo “como”, que pode ser um advérbio de modo ou uma inflexão do verbo “comer”. Desse modo, a lematização sobre a seguinte frase “Eu como muitas frutas, como laranjas e morangos”, gera o lema “como” para o advérbio e o lema “comer” para o verbo.

Diante do exposto e sabendo que o grupo de documentos dos dados utilizados nesse projeto não é extenso, cerca de 57 ementas, priorizou-se a acurácia em vez do desempenho. Por conseguinte, optou-se pelo uso da lematização e da biblioteca do spaCy, a qual fornece uma ferramenta que faz a análise estrutural dos textos e também possui um lematizador.

C. Treinamento

Essa etapa da implementação consiste na realização de uma série de manipulações matemáticas em matrizes, as quais modelam os dados até que se obtenha valores que possuam significado e possam gerar os resultados dos mapeamentos. Assim, o primeiro passo para ser capaz de fazer as operações algébricas é quantificar as informações armazenadas nos documentos. Isso é feito utilizando a técnica do *Bag-of-Words*, que, para toda palavra encontrada em um documento, cria-se um identificador para esse termo e se realiza a contagem da frequência com que essa palavra aparece naquele documento [34]. Cada documento passa a ser definido, então, por um conjunto de identificadores e suas respectivas frequências de ocorrência.

Sabendo disso, o *Bag-of-Words* consegue sintetizar todo o corpo de documentos para uma matriz em que há um eixo para os documentos e outra para cada palavra distinta, sendo que cada elemento dessa matriz possui um valor numérico que representa a respectiva frequência dessa palavra naquele documento. Essa matriz é conhecida como a matriz termo-documento e um exemplo dela pode ser visto na Figura 1.

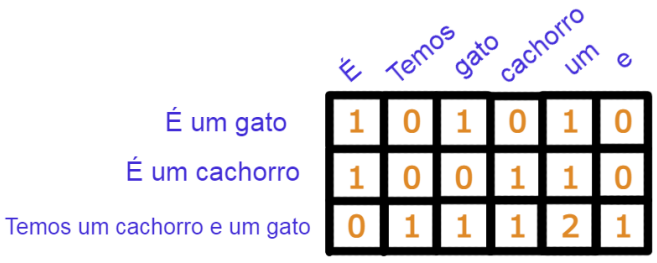

Figura 1. Representação de uma matriz do tipo termo-documento

Quantificados os textos para uma matriz, realiza-se a primeira manipulação matricial do *Term frequency – inverse document frequency* (TF-IDF) com o objetivo de elevar a acurácia [35]. Essa operação é uma transformada que tem como entrada a matriz termo-documento e aplica-se sobre todas as frequências a manipulação do TF-IDF, gerando uma nova matriz termo-documento com esse novos valores. Essa operação torna as palavras raras do corpo de documentos mais proeminentes e efetivamente faz com que as palavras mais comuns se tornem menos relevantes [35]. Formalmente, os valores iniciais das frequências relativas são multiplicados pela razão normalizada entre a quantidade total de documentos do corpo pela quantidade de documentos que contém essa certa

palavra. Geralmente, a operação de normalização é feita pelo operador logarítmico, mas se pode utilizar desde o cosseno ou até mesmo funções personalizadas [36]. A equação específica utilizada nesse projeto pode ser vista na Equação 1 [37], em que *NovoValor* representa o valor do TF-IDF; *FreqRelativa* é a frequência da palavra i no documento j; *DocsTotal* é a quantidade total de documentos e *DocsContem* é a quantidade de documentos que contém a palavra i.

$$NovoValor = FreqRelativa_{ij} \cdot \log_2 \left(\frac{DocsTotal}{DocsContem_i} \right) \quad (1)$$

A seguir, fez-se a principal manipulação dos valores, que é o método do LSI [11], que, relembrando, se baseia na ideia de que documentos com significados relacionados tendem a possuir palavras em comum. Basta, então, conhecer a frequência de ocorrência de cada palavra e normalizar sinônimos e inflexões para poder comparar documentos entre si. Desse modo, para encontrar documentos semelhantes, o LSI faz uma decomposição matricial conhecida como *Singular Value Decomposition* (SVD), que transforma uma matriz do tipo termo-documento em três diferentes matrizes menores: U, Σ, V^T.

A U (*left singular vectors*) é a matriz cujos elementos possuem a relação de similaridade entre termos e tópicos latentes. A Σ (*Singular values matrix*) consiste na matriz quadrada diagonal que armazena na diagonal principal e, em ordem de importância, os tópicos latentes encontrados no corpo de documentos. Finalmente, a V^T (*right singular vectors*) é constituída pelas similaridades entre documentos e tópicos latentes.

Para a execução desse algoritmo, utilizou-se novamente a biblioteca do Gensim, que implementa uma versão rápida e truncada do SVD. Isso significa que essa versão retira os tópicos latentes menos importantes e, assim, diminui a dimensionalidade das matrizes, melhorando o desempenho do algoritmo e economizando memória. A decomposição do SVD pode ser vista na Figura 2, na qual A é a matriz termo-documento, n é o número de documentos, m é o número de palavras ou termos distintos, e r é o número de conceitos ou tópicos latentes encontrados em todo o corpo de documentos.

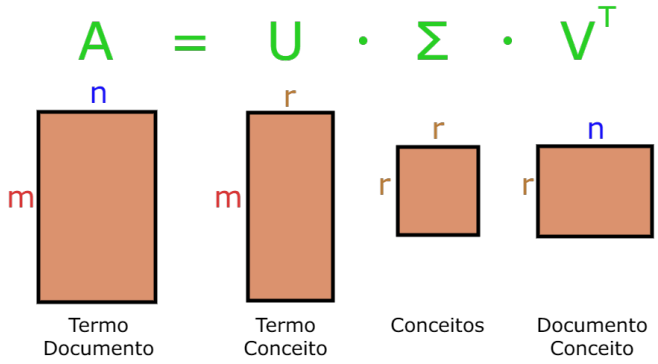

Figura 2. Representação da decomposição SVD

D. Buscas por similaridade

Finalmente, para relacionar documentos similares, pode-se utilizar a matriz V^T, que contém as proximidades de todos os documentos com os tópicos latentes encontrados no corpo. Com essa matriz, é possível comparar indiretamente um novo documento com cada um dos demais documentos utilizados no treinamento a partir dos valores na matriz. Voltando para o problema principal de mapeamento das disciplinas com a legislação, pode-se realizar o treinamento sobre o corpo de documentos das disciplinas e, em seguida, classificar um novo documento contendo um conteúdo técnico nas categorias das ementas.

Formalmente, esse novo documento que contém um conhecimento técnico e não foi utilizado no treinamento anterior pode ser pré-processado do mesmo modo. No entanto, as demais etapas são distintas ao que foi feito. Esse documento deve ser quantificado e processado pelo TF-IDF só considerando os termos já obtidos pelo modelo anterior e o algoritmo do LSI também deve considerar apenas os tópicos latentes já definidos. Após tudo isso, converte-se o novo documento em um vetor x de tamanho r, cujos valores representam a proximidade com cada uns dos r tópicos latentes já previamente definidos pelo corpo de documentos do treinamento. Desse modo, esse novo vetor contém o significado geral do documento dividido entre os r tópicos latentes.

Da mesma forma, a matriz V^T, possui, em cada vetor linha que a compõe, a mesma divisão do conceito geral entre os tópicos latentes para todo os n documentos utilizados no treinamento. Sabendo disso, pode-se fazer o produto entre a matriz V^T, de dimensões $\{n, r\}$, e o vetor x, de dimensões $\{r, 1\}$, para obter um vetor resultante y, de dimensões $\{n, 1\}$. Essa operação matricial pode ser vista na Figura 3.

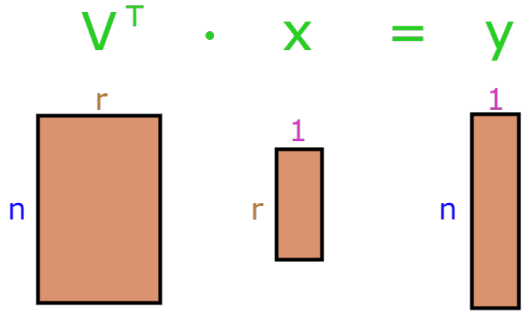

Figura 3. Representação do produto entre a matriz documento-conceito e o vetor do novo documento, que resulta no vetor de similaridades

O vetor y obtido possui em seus elementos a relação de similaridade entre cada um dos n documentos utilizados no treinamento e o novo documento. Isso porque esses novos valores obtidos são os somatórios de todos os produtos entre a proximidade de um documento do treinamento e a proximidade do novo documento para cada tópico latente.

Para cada conceito, se ambos os documentos considerados tiverem um valor positivo elevado, indicando que ambos são

muito próximos desse tópico e, portanto, possuem uma relação de sinonímia, o produto entre eles aumenta consideravelmente o valor da soma final, que representa a similaridade geral entre os documentos. Da mesma forma, se ambos tiverem valores negativos elevados, indicando uma relação de antonímia de ambos os documentos com o tópico latente, o produto também gera um valor positivo grande, o qual será somado à similaridade, aumentando-a.

Por outro lado, se os valores tiverem sinais contrários, indicando que um é próximo ao tópico latente e outro é distante, soma-se um valor negativo à similaridade, decrescendo-a. Ao fim, se os valores forem próximos de 0, indicando indiferença ao tópico latente, pouco se alterará na soma e, portanto, na similaridade. Os comportamentos desses casos podem ser vistos na Figura 4.

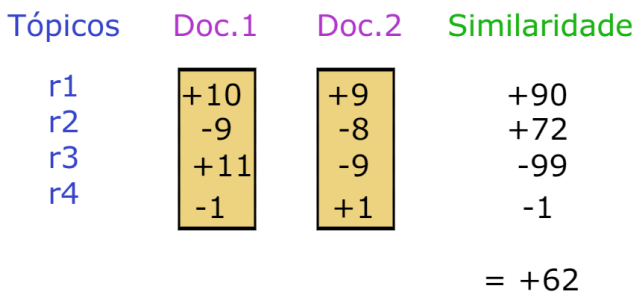

Figura 4. Ilustração dos casos possíveis de relação entre proximidades ou similaridades por tópico latente

Diante de todas as somas finais das similaridades gerais entre dois documentos, uma etapa final de normalização é realizada para limitar o valor ao intervalo contínuo $[-1, 1]$. Essa operação de normalização consiste em dividir todos os valores de similaridade pelo produto das normas dos respectivos vetores considerados. Desse modo, toda essa operação pode se resumir ao cálculo do cosseno entre os vetores, que tem o contradomínio definido em $[-1, 1]$. Por esse motivo, essa similaridade obtida entre os documentos também é chamada de similaridade de cosseno. Essa etapa de normalização é feita para comparar as diferentes similaridades obtidas com uma mesma métrica.

Desse modo, sabe-se que o valor -1 tem o significado de máxima antonímia, indicando que os documentos possuem significados contrários. Por outro lado, o valor de 1 indica máxima sinonímia, ou seja, os documentos são semanticamente idênticos. Além disso, o valor neutro de 0 indica indiferença, negando uma relação de proximidade entre os documentos.

Analogamente ao que foi feito, pode-se fazer o mapeamento inverso, classificando as disciplinas nos conteúdos técnicos ao utilizar as ementas como os novos documentos. Nota-se, então, a versatilidade do artefato desenvolvido, o qual também pode ser expandido para outros cursos ou até mesmo para outras instituições de ensino superior, requerendo apenas que se saiba as ementas do curso.

Nesse trabalho, para saber quais são os conteúdos técnicos, utilizou-se a planilha "Matriz do Conhecimento", que consiste em uma tabela que sintetiza quais são os conhecimentos técnicos exigidos para cada competência do CONFEA e também quais competências são contempladas por cada tipo de engenharia. Esse documento foi adquirido a partir de um ex-conselheiro do CREA e a sua utilização está sendo oficialmente discutida pela autarquia.

IV. Resultados

A interface escolhida para interação do programa com o usuário foi a linha de comando, implementada utilizando o *Argparse* [10], uma biblioteca nativa do Python, que foi configurada para receber as informações pela linha de comando e executar as buscas de similaridade. Cada busca de similaridade consiste em chamar o programa juntamente com o texto de um novo documento fornecido pelo usuário para que o programa retorne uma listagem das ementas com os maiores valores de similaridade, variando de 0 a 1, ignorando os valores negativos de antonímia.

Finalmente, baseou-se nos conteúdos técnicos atribuídos ao curso de Engenharia Mecatrônica, que estão listados na "Matriz do Conhecimento" no setor de "Controle e Automação", para gerar os resultados. O formato dessas buscas consiste no executável do programa, nomeado como "creatools.py", o curso utilizado como referencial e fornecendo o texto do novo documento depois da opção $-q$ (*query*). Alguns exemplos serão exibidos a seguir.

```
> python creatools.py mecatronica -q Cálculo
diferencial e integral

0.72 CÁLCULO 3
0.64 CÁLCULO 1
0.58 CÁLCULO 2
0.26 VARIAVEL COMPLEXA 1
0.22 SINAIS E SISTEMAS EM TEMPO CONTÍNUO

> python creatools.py mecatronica -q Mecânica

0.55 MECANICA 1
0.53 FISICA 2
0.50 MECANICA 2
0.39 FISICA 1
0.35 DESENHO MECANICO ASSISTIDO POR
COMPUTADOR 1
```

Nesses dois primeiros exemplos, observa-se que o programa conseguiu com sucesso encontrar as principais matérias que envolvem os tópicos informados, gerando altos valores de similaridade para as ementas que possuem em seu documento uma frequência elevada de ocorrências das palavras pesquisadas, e valores mais baixos para frequências menores. A ementa de **Cálculo 3**, por exemplo, possui 24 ocorrências dos termos buscados enquanto a ementa de **Sinais e Sistemas em Tempo Contínuo** possui apenas 5.

[10]https://docs.python.org/3/library/argparse.html

Além disso, vale ressaltar que as inflexões das palavras buscadas foram unificadas pelo lema, o que auxiliou o programa a identificar similaridades entre as ementas que apresentam a palavra "mecânica" ou "mecânico" por exemplo. Isso pode ser observado pelo resultado de **Desenho Mecânico Assistido por Computador 1** gerado ao buscar o termo "Mecânica".

```
> python creatools.py mecatronica -q Forças
Intermoleculares

0.59 QUIMICA GERAL TEORICA
0.51 FISICA 1
0.47 MECANICA 1
0.37 FISICA 2
0.34 FISICA 3 EXPERIMENTAL
```

Nessa terceira busca, observa-se que o texto "Forças Intermoleculares", tópico de estudo do ramo da Química, gerou a ementa de **Química Geral Teórica**. Não obstante, também encontrou-se resultados da Física devido à alta frequência de ocorrência do termo "forças", assunto muito estudado nas matérias da Física e, portanto, aparece com muita frequência nas ementas.

```
> python creatools.py mecatronica -q Matemática

0.74 CONTROLE NO ESPAÇO DE ESTADOS
0.54 CONTROLE PARA AUTOMAÇÃO
0.28 ORGANIZAÇÃO INDUSTRIAL
0.00 INTRODUÇÃO AOS CIRCUITOS ELÉTRICOS
0.00 INTRODUCAO A ALGEBRA LINEAR
```

No entanto, o algoritmo do LSI adotado tem suas limitações. Na busca pelo ramo do conhecimento da matemática, o programa não conseguiu identificar algumas das disciplinas relacionadas. Até mesmo matérias que são diretamente do ramo da matemática, como as de **Cálculo** e de **Variável Complexa**, não foram encontradas. Isso se deve ao fato de que as ementas dessas disciplinas não incluem essa palavra diretamente.

```
> python creatools.py mecatronica -q Estrutura
de Dados

0.69 TRANSMISSÃO DE DADOS
0.49 ALGORITMOS E PROGRAMAÇÃO DE COMPUTADORES
0.31 ESTRUTURAS DE DADOS
0.29 INSTRUMENTACAO DE CONTROLE
0.26 CONTROLE DIGITAL
```

Ademais, a busca por "Estrutura de Dados", tema de estudo da área de conhecimento de *Algoritmos e Estruturas de Dados*, gerou um valor de relevância maior para a disciplina de Transmissão de Dados do que para as matérias que realmente estudam esse tópico, como **Algoritmos e Programação de Computadores** e a própria disciplina de **Estrutura de Dados**. Esse comportamento é causado pelo modo como foi escrita a ementa de **Transmissão de Dados**, a qual possui 12 ocorrências da palavra "dados", o que acabou elevando sua relevância e ofuscando as matérias mais relevantes.

Como se pode observar pelos resultados obtidos, o modelo do LSI consegue encontrar o significado de um documento e mapeá-lo a outros documentos semelhantes. No entanto, esse funcionamento é dependente das palavras que cada ementa contém e, portanto, bastante influenciado pelo modo como foram escritos os documentos, o que pode prejudicar a qualidade dos resultados.

Além disso, dificultando ainda mais, o método de normalização da lematização não é perfeito e pode deixar passar algumas inflexões ou convertê-las para lemas diferentes, mesmo quando o desejo é de agregá-las. Isso pode ser visto na conversão feita pelo programa para a palavra "série", que se tornou o verbo "seriar" em alguns casos em vez do lema "série". Outro exemplo são as palavras "eletromagnetismo" e "eletromagnético" que possuem o mesmo significado, mas não foram agregadas por serem de diferentes classes gramaticais. Desse modo, os problemas de normalização encontrados foram corrigidos com um dicionário manualmente definido, porém algumas conversões incorretas ainda não foram identificadas e, portanto, persistem, prejudicando a acurácia do programa.

Não obstante, os resultados obtidos são bem satisfatórios e podem ser utilizados sem dificuldades por qualquer usuário. A administração da UnB, por exemplo, já pode utilizar-se desse artefato para averiguar como os seus projetos pedagógicos atuais de curso se relacionam com a legislação vigente. Sabendo disso, implementou-se também no programa algumas funcionalidades a mais para melhorar a experiência do usuário, como a opção de fazer múltiplas buscas sem precisar executar novamente o programa, definir quantos resultados por busca devem aparecer, escolher o valor mínimo de similaridade aceito ou até mesmo um programa que faz o mapeamento inverso.

V. Conclusão

Considerando a legislação vigente, que exige uma gama de conteúdos técnicos, os quais são abrangidos por diversas competências. Uma ferramenta de mapeamento das disciplinas de um curso com os conteúdos exigidos é de interesse para diversos usuários. Pode-se citar, por exemplo, os estudantes de engenharia e as instituições de ensino superior que ofertam cursos abrangidos por essa legislação, como a UnB. Desse modo, os alunos conseguem facilmente, a partir do programa desenvolvido, pesquisar quais disciplinas devem estudar para adquirir as competências necessárias para o mercado de trabalho desejado. Por outro lado, as instituições de ensino, conhecendo bem o mapeamento dos conteúdos, estão habilitados para reorganizar os projetos pedagógicos de seus cursos, abrangendo todos os requerimentos do CREA

Desse modo, utilizando-se de algoritmos da área de PLN, como o LSI, e também da linguagem de programação Python e as bibliotecas disponíveis, criou-se essa ferramenta de mapeamento. Ademais, utilizou-se uma série de métodos e modelos auxiliares para pré-processar os dados, como a lematização e o TF-IDF, adaptando o corpo de documentos para um formato de fácil processamento para o LSI. Ao fim, com o intuito

de simplicidade, adotou-se o *Argparse* como interface para interação com os usuários.

Os resultados obtidos demonstram que o programa consegue resolver o problema proposto, que é mapear as disciplinas de um curso com os conhecimentos técnicos exigidos pelas competências do sistema CREA/CONFEA, caso seja fornecido os documentos necessários. Para os testes desse programa, adotou-se o fluxo da Engenharia Mecatrônica da UnB no treinamento do modelo de LSI e, para fazer o mapeamento, foi informado os conhecimentos técnicos em formato de texto para as buscas de similaridades da ferramenta.

Não obstante o funcionamento do programa, ele possui certas limitações que podem ser resolvidos em trabalhos futuros, como elaborar um algoritmo melhor para o pré-processamento dos dados, extraindo com mais efetividade as informações contidas. Isso pode ser feito com uma etapa de limpeza dos dados ou de normalização mais acuradas, por exemplo. Além disso, pode-se também utilizar outros modelos de processamento de linguagem natural ou aprendizagem de máquina que possuam melhor desempenho e/ou acurácia para esse problema de mapeamento.

Referências

[1] "Resolução nº 2, de 18 de junho de 2017." http://portal.mec.gov.br/cne/arquivos/pdf/2007/rces002_07.pdf. Acessado em: 26 de Abril de 2022.

[2] "Referenciais nacionais dos cursos de engenharia." http://portal.mec.gov.br/dmdocuments/referenciais2.pdf. Acessado em: 26 de Abril de 2022.

[3] MEC/CES, "Resolução nº 2, de 24 de abril de 2019," tech. rep., Ministério da Educação, 2019. Acessado em: 26 de Abril de 2022.

[4] "ResoluÇÃo n° 1.073, de 19 de abril de 2016." https://normativos.confea.org.br/Ementas/Visualizar?id=59111. Acessado em: 11 de Julho de 2022.

[5] "Resolução nº 1, de 26 de março de 2021." http://portal.mec.gov.br/index.php?option=com_docman&view=download&alias=175301-rces001-21&category_slug=marco-2021-pdf&Itemid=30192. Acessado em: 26 de Abril de 2022.

[6] E. Cambria and B. White, "Jumping nlp curves: A review of natural language processing research [review article]," *IEEE Computational Intelligence Magazine*, vol. 9, no. 2, pp. 48–57, 2014.

[7] IBM, "Natural language processing (nlp)." Acessado em: 12 de Julho de 2022.

[8] S. P. and A. P. Shaji, "A survey on semantic similarity," in *2019 International Conference on Advances in Computing, Communication and Control (ICAC3)*, pp. 1–8, 2019.

[9] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Information*, vol. 10, no. 4, 2019.

[10] Y. Wan and H. Tong, "Categorization and monitoring of internet public opinion based on latent semantic analysis," in *2008 International Seminar on Business and Information Management*, vol. 2, pp. 121–124, 2008.

[11] M. W. Berry, S. T. Dumais, and G. W. O'Brien, "Using linear algebra for intelligent information retrieval," *SIAM Rev.*, vol. 37, pp. 573–595, 1995.

[12] P. Kherwa and P. Bansal, "Latent semantic analysis: An approach to understand semantic of text," in *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)*, pp. 870–874, 2017.

[13] K. Smelyakov, D. Karachevtsev, D. Kulemza, Y. Samoilenko, O. Patlan, and A. Chupryna, "Effectiveness of preprocessing algorithms for natural language processing applications," in *2020 IEEE International Conference on Problems of Infocommunications. Science and Technology (PIC S&T)*, pp. 187–191, 2020.

[14] S. Raschka, J. Patterson, and C. Nolet, "Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence," *Information*, vol. 11, no. 4, 2020.

[15] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*. O'Reilly, 01 2009.

[16] O. M. Toledo, "A estrutura curricular do curso de engenharia de controle e automaÇÃo do cefet-mg concebida por eixos de conteÚdos e atividades," *Anais do XXXIV COBENGE*, 2006.

[17] L. Rezende, L. Segre, G. Campos, and B. Campos, "Mapeamento de competências através da análise curricular de um curso de ciência da computação," *Anais do XXVI Congresso da SBC, XIV Workshop sobre Educação em Computação*, 01 2006.

[18] J. Hoblos, "Experimenting with latent semantic analysis and latent dirichlet allocation on automated essay grading," in *2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pp. 1–7, 2020.

[19] A. A. P. Ratna, R. Sanjaya, T. Wirianata, and P. Dewi Purnamasari, "Word level auto-correction for latent semantic analysis based essay grading system," in *2017 15th International Conference on Quality in Research (QiR) : International Symposium on Electrical and Computer Engineering*, pp. 235–240, 2017.

[20] M. Alsallal, R. Iqbal, S. Amin, and A. James, "Intrinsic plagiarism detection using latent semantic indexing and stylometry," in *2013 Sixth International Conference on Developments in eSystems Engineering*, pp. 145–150, 2013.

[21] B. Giesbers, E. Rusman, and J. Bruggen, "State of the art lsa," *Collaborative Open Environment for Project-Centred Learning*, 05 2006.

[22] K. Demertzis, "Data preprocessing," 07 2019.

[23] Cognilytica, "Data preparation & labeling for ai 2020." Acessado em: 25 de Junho de 2022.

[24] K. V. Ghag and K. Shah, "Comparative analysis of effect of stopwords removal on sentiment classification," in *2015 International Conference on Computer, Communication and Control (IC4)*, pp. 1–6, 2015.

[25] I. A. Braga, "Evaluation of stopwords removal on the statistical approach for automatic term extraction," in *2009 Seventh Brazilian Symposium in Information and Human Language Technology*, pp. 142–149, 2009.

[26] A. Wibowo Haryanto, E. Kholid Mawardi, and Muljono, "Influence of word normalization and chi-squared feature selection on support vector machine (svm) text classification," in *2018 International Seminar on Application for Technology of Information and Communication*, pp. 229–233, 2018.

[27] D. Khyani and S. B S, "An interpretation of lemmatization and stemming in natural language processing," *Shanghai Ligong Daxue Xuebao/Journal of University of Shanghai for Science and Technology*, vol. 22, pp. 350–357, 01 2021.

[28] W. R. Merrifield, "P. h. matthews, inflectional morphology: a theoretical study based on aspects of latin verb conjugation, (cambridge studies in linguistics, 6.) cambridge: Cambridge university press, 1972. pp. x 431.," *Journal of Linguistics*, vol. 10, no. 2, p. 349–355, 1974.

[29] E. GmbH, "Trained models & pipelines." Acessado em: 12 de Julho de 2022.

[30] M. Porter, "An algoritm for suffix stripping," *Program*, vol. 14, pp. 130–137, 07 2006.

[31] V. Orengo and C. Huyck, "A stemming algorithm for the portuguese language," in *Proceedings Eighth Symposium on String Processing and Information Retrieval*, pp. 186–193, 2001.

[32] G. Nicolai and G. Kondrak, "Leveraging inflection tables for stemming and lemmatization.," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 1138–1147, Association for Computational Linguistics, Aug. 2016.

[33] V. Balakrishnan and E. Lloyd-Yemoh, "Stemming and lemmatization: A comparison of retrieval," tech. rep., University of Malaya, Kuala Lumpur, Malaysia, 2014. Acessado em: 25 de Junho de 2022.

[34] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: A statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, pp. 43–52, 12 2010.

[35] M. Das, S. Kamalanathan, and P. Alphonse, "A comparative study on tf-idf feature weighting method and its analysis using unstructured dataset," in *COLINS*, 2021.

[36] G. R. Rehurek, "Tf-idf model." Acessado em: 13 de Julho de 2022.

[37] P. Jotikabukkana, V. Sornlertlamvanich, O. Manabu, and C. Haruechaiyasak, "Effectiveness of social media text classification by utilizing the online news category," in *2015 2nd International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, pp. 1–5, 2015.