Processamento de dados textuais: aplicação da biblioteca NLTK como ferramenta analítica.

Vinícius Andrade Lopes[1]*; João Vitor Matos[2]

[1] Faculdade Rede de Ensino DOCTUM. Bacharel em Sistemas de Informação. Rua Uberlândia, 331 – Centro; 35160-024 Ipatinga, MG, Brasil.
[2] Universidade Federal Fluminense (UFF). Mestre em Economia. Rua Miguel de Frias, 9 – Icaraí; 24220-900 Niterói, RJ, Brasil
*autor correspondente: vinicius.andlopes@gmail.com

Processamento de dados textuais: aplicação da biblioteca NLTK como ferramenta analítica.

Resumo

A evolução tecnológica tem sido um fator crucial para o desenvolvimento de diversas áreas no mundo, como setores econômicos, sociais, comércios e outros segmentos. Com o aumento da inclusão digital, a quantidade de informações geradas e disponibilizadas na internet tem crescido de maneira exponencial, o que tem impulsionado o desenvolvimento de tecnologias capazes de extrair conhecimento a partir desses dados. Nesse contexto, o processamento de linguagem natural surge como uma área da inteligência artificial amplamente utilizada para coletar informações dos dados textuais disponibilizados em diversas bases, visando compreender a natureza desses dados e qual o seu impacto no processo de tomada de decisões. Essa técnica de extrair informações por meio de dados textuais tem sido utilizada em diversas áreas, desde a análise de sentimentos em redes sociais até o processamento de grandes quantidades de dados em indústrias, e tem se mostrado uma ferramenta poderosa para análise de textos literários. Com base nisso, este projeto tem como objetivo analisar os livros e os filmes da saga Harry Potter por meio do processamento de linguagem natural, a fim de extrair informações relevantes para aprimorar o processo de leitura, análise literária e compreensão textual. Para alcançar esse propósito, os dados dos “datasets” foram coletados e submetidos a técnicas de pré-processamento como limpeza dos dados, remoção de palavras irrelevantes, tokenização e normalização, visando padronizar as informações e permitir uma análise mais eficiente.

Palavras-chave: processamento de linguagem natural; inteligência artificial; pré-processamento de dados; visualização de dados; tokenização.

Resumen

La evolución de la tecnología ha sido un factor de mucha importancia para el desarrollo de varias áreas del mundo, como, por ejemplo, sectores económicos, sociales, comerciales y otros segmentos. Con el crecimiento de la inclusión digital, la cantidad de informaciones generadas y que están disponibles en internet han aumentado de manera exponencial, lo que ha impulsado el desarrollo de tecnologías capaces de extraer conocimiento de esos datos. En ese contexto, el procesamiento del lenguaje natural surge como un área de la inteligencia artificial ampliamente utilizada para recompilar información de los datos textuales disponibles en diversas fuentes, para comprender la naturaleza de estos datos y su impacto en el proceso de toma de decisiones. Esta técnica de obtener información a través de datos textuales se ha utilizado en varios campos, desde el análisis de sentimientos en las redes sociales hasta el

procesamiento de grandes cantidades de datos industriales, y ha demostrado ser una poderosa herramienta para el análisis de textos literarios. En ese caso, este proyecto tiene como objetivo analizar los libros y las películas de la saga Harry Potter por medio del procesamiento de lenguaje natural, con el fin de extraer informaciones relevantes para mejorar el proceso de lectura, análisis literario y comprensión textual. Para alcanzar ese propósito, los datos de los "datasets" fueron colectados y sometidos a técnicas de preprocesamiento como limpieza de datos, remoción de palabras irrelevantes, tokenización y normalización, con el objetivo de estandarizar la información y permitir un análisis más eficiente.

Palabras-clave: procesamiento del lenguaje natural; inteligencia artificial; preprocesamiento de datos; visualización de datos; tokenización.

Introdução

Com o advento da tecnologia e o grande número de pessoas conectadas à internet, é inegável que a quantidade de informações trafegadas pelos principais meios de comunicação, tem se convertido em uma base de dados mundial não estruturada para empresas e profissionais, que possuem um grande potencial técnico e analítico, coletarem informações relacionadas a diversos seguimentos. No entanto, extrair informações robustas e conclusivas dessa base mundial de dados tem se tornado uma tarefa complexa, e alguns aspectos devem ser levados em consideração.

As dificuldades a serem encontradas durante a interpretação de dados textuais podem ser diversas, e elas ocorrem basicamente porque existem várias características textuais que precisam ser compreendidas. Os livros são classificados por gêneros literários, e a alteração no estilo e no conceito de escrita é bastante comum quando se compara um livro de fantasia com um romance, por exemplo. Alguns livros podem conter até mesmo gírias específicas, a variar da região de origem. Já as postagens feitas em redes sociais, são normalmente escritas de maneira informal e com a utilização de abreviações, caracteres especiais e emoticons.

Em complemento à dificuldade na compreensão dos dados, é possível se deparar com outro ponto que deve ser considerado durante o desenvolvimento de análises textuais: o idioma. Atualmente, existem mais de 6.500 idiomas falados em todo o mundo e, dentre eles, os mais populares são o Inglês, Chinês (Mandarim), Hindi, Espanhol e Francês (Gazeau, 2018). Dessa forma, identificar o idioma correto para atender as necessidades da análise textual, pode facilitar todo o processo de coleta de informações e otimizar a capacidade de processamento do "hardware", visto que filtrar toda a base de dados a apenas um idioma, por exemplo, requer menos poder computacional.

Executar tarefas manuais para extração de informações em repositórios contendo um grande volume de dados não estruturados, ou estruturados, é uma tarefa praticamente impossível de ser realizada. Entretanto, não basta somente ter acesso a grandes volumes de dados. O ponto chave a ser realmente considerado, é como encontrar uma informação útil dentro de um arcabouço de dados diversificados (Yang et al., 2020). Desenvolver uma base de dados estruturada, sem erros de formatação e com os devidos parâmetros definidos, facilita a objetividade da visualização e análise das informações (Wickham, 2016).

Sendo assim, existe uma área da computação que tem por objetivo extrair representações e significados completos de textos escritos de forma livre em linguagem natural, denominada "Natural Language Processing" [NLP] ou Processamento de linguagem natural [PLN], em português (Indurkhya, 2010). Entende-se por linguagem natural os meios de comunicação mais comuns entre seres-humanos, como os próprios idiomas citados anteriormente. Seu crescimento e evolução acontece de geração em geração, e é difícil descrever linguagens naturais por completo seguindo preceitos explícitos, diferentemente de linguagens de programação e fórmulas matemáticas, que possuem princípios já estabelecidos de suas definições (Bird et al., 2009).

Basicamente, a PLN utiliza preceitos linguísticos como classe de palavras para realizar as análises, como por exemplo substantivos, verbos, adjetivos, pronomes, dentre outros, além de diversas estruturas gramaticais que têm por objetivo dar sentido às sentenças analisadas. Isso ocorre em função das várias representações de conhecimento, como um conjunto de palavras existentes em um idioma e seus significados, propriedades e regras gramaticais da linguagem, um grande vocabulário de palavras com relações semânticas, sinônimos e abreviações, e ontologias de entidade e ações (Indurkhya, 2010). O processamento de linguagem natural abrange qualquer tipo de desenvolvimento computacional de linguagens naturais, e consiste em compreender conteúdos descritos por humanos até o ponto de fornecer uma resposta válida sobre a informação analisada (Bird et al., 2009).

O processamento de linguagem natural pode ser definido, de forma simplificada, como uma modelo para identificar quem fez o quê, a quem, quando, onde, como e por que (Robertson, 1946). A PLN considera os textos como uma sequência de caracteres, respeitando a estrutura hierárquica da linguagem que está sendo analisada. Sendo assim, as técnicas de processamento de linguagem natural podem ser utilizadas em “softwares” de diversos seguimentos, como por exemplo corretores gramaticais, conversores de fala para textos, aplicações capazes de traduzir textos para outros idiomas, análise de sentimentos dos usuários mediante a um tema, dentre outros aspectos (Indurkhya, 2010).

No presente trabalho, serão utilizadas técnicas de processamento de linguagem natural nos “datasets” dos livros e filmes da saga Harry Potter. O objetivo é extrair informações

relevantes, como frequência de palavras e padrões de linguagem, que possam auxiliar na análise comparativa entre as versões literárias e cinematográficas. O uso dessas técnicas permitirá uma análise mais aprofundada dos dados, possibilitando uma compreensão mais ampla e precisa das diferenças e semelhanças entre as duas mídias.

Material e Métodos

A estratégia de pesquisa utilizada no desenvolvimento deste projeto será exploratória, visando apresentar de forma clara e concisa, didática e prática, a implementação das técnicas de processamento de linguagem natural disponibilizadas pela biblioteca “Natural Language Toolkit” [NLTK][1]. Descrever detalhadamente as principais etapas do processamento de textos de linguagens naturais é crucial para a compreensão do funcionamento analítico da ferramenta. As etapas de análises do PLN são decompostas em estágios, conforme apresentados na Figura 1.

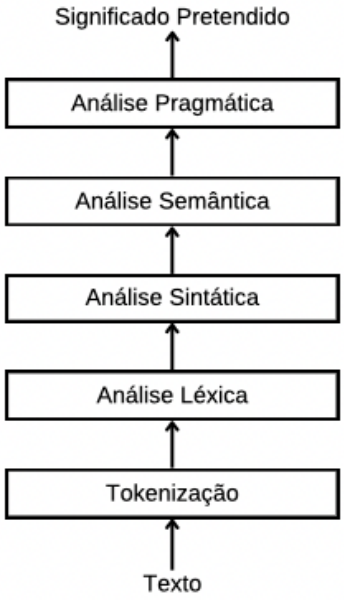

Figura 1. Estágios de análise do processamento de linguagem natural
Fonte: Adaptado de Dale et al. (2000)

A execução da etapa de tokenização é um processo extremamente importante na análise de linguagem natural, pois permite manipular conteúdos textuais de maneira mais eficiente. O conceito que engloba essa técnica, segundo Castro (2017), é basicamente uma operação que é capaz de dividir cada sentença de um texto em uma sequência de tokens,

[1] Página oficial: http://nltk.org/

onde cada token é uma unidade léxica como, por exemplo, uma palavra ou frase. Jurafsky (2000) complementa que a tokenização é o processo capaz de realizar um “split” de textos em palavras, frases, símbolos, ou qualquer outro conjunto de caracteres significativos, conhecido como tokens, que podem ser usados posteriormente para uma análise mais robusta. A Figura 2 exemplifica como funciona o processo de tokenização na prática.

```
# Frase fornecida como "input"
_str = 'Harry Potter é a melhor saga de todas!'

# Função em python para realizar a tokenização, utilizando a
↪  biblioteca NLTK
print(nltk.tokenize.word_tokenize(_str))

# Tokenização fornecida como "output"
['Harry', 'Potter', 'é', 'a', 'melhor', 'saga', 'de', 'todas', '!']
```

Figura 2. Exemplo prático de como funciona a etapa de tokenização de uma frase
Fonte: Dados originais da pesquisa

Existem diversos métodos de executar essa técnica de tokenização, onde cada uma possui suas vantagens e desvantagens. Um método bastante comum na divisão de textos é a utilização de espaços vazios nas sequências de caracteres, onde ocorre a geração desses tokens textuais a cada espaço vazio encontrado (Jurafsky, 2000). No entanto, usar esse tipo de método para a tokenização de textos pode ser complicada em situações em que o espaço é utilizado de forma inconsistente, como por exemplo em frases escritas no idioma inglês que contenham algumas abreviações, como “Mr.” (Kilgarriff and Grefenstette 2003).

Outro método bastante comum é a tokenização baseada em expressões regulares. Basicamente, esse tipo de abordagem utiliza um conjunto de regras para identificar e separar os textos, tornando a etapa de tokenização mais assertiva quando comparado ao método baseado em espaços vazios, porém aumentando o nível de complexidade durante a implementação algorítmica (Chang et al., 2014). Existe também um método de tokenização baseada em aprendizado de máquina, onde um algoritmo é treinado com exemplos de textos já tokenizados e, a partir desse modelo já treinado, o sistema é capaz de identificar qual a melhor forma de realizar a tokenização do texto fornecido como “input”. Esse último método citado é mais eficiente e o mais complexo de ser implementado, pois requer um alto nível de conhecimento sobre o tema, sendo necessário desenvolver o conjunto de dados de treinamento ou possuir um conjunto com os parâmetros já treinados (Jiang et al., 2017).

A compreensão dos elementos linguísticos é uma etapa crucial no processamento de linguagem natural, onde é possível identificar e categorizar as palavras, símbolos e marcas

de pontuações presentes em um texto. Essa tarefa é realizada por um processo chamado analisador léxico. De acordo com Jurafsky (2020), o analisador léxico é responsável por ler o texto, caractere por caractere, e gerar uma série de tokens, tais como palavras-chave, identificadores, números e símbolos, que serão utilizados posteriormente pelo sistema.

A etapa de análise léxica é fundamental para que o computador possa interpretar os dados textuais com alta precisão. Por exemplo, ao reconhecer uma palavra como um verbo, o sistema pode entender que essa palavra se relaciona a uma ação, e, portanto, pode ser utilizada para realizar certas tarefas. Além disso, a realização da etapa de análise léxica é essencial para o bom desempenho da etapa de análise sintática, pois os tokens gerados pelo analisador léxico serão usados como entrada para outros componentes do processamento de linguagem natural (Jurafsky, 2000).

A análise sintática é uma fase da etapa de análise léxica, na qual fica responsável por identificar e analisar a estrutura gramatical de uma frase. Essa técnica é usada para determinar as funções sintáticas de cada palavra dentro de uma frase, buscando relações entre essas palavras (Chomsky, 2009). De forma similar as etapas descritas anteriormente, existem diversos métodos de realizar a análise sintática, dentre eles: gramáticas formais, análise estatística e análise de dependência.

As gramáticas formais, de forma resumida, são responsáveis pelo entendimento estrutural de cada frase fornecida como “input” para o algoritmo, identificando o contexto e a relevância de cada palavra (Chomsky, 2009). Já a análise estatística se baseia em modelos estatísticos para compreender a estrutura gramatical de uma frase a partir de dados de treinamento. Esses modelos têm capacidade de lidar com a ambiguidade e a variação na língua natural, se tornando mais robustos em relação aos erros e ao ruído (Charniak, 1997). A análise de dependência se concentra na relação de dependência entre as palavras de uma frase, ou seja, essa técnica tem como princípio de que as palavras de uma frase estão relacionadas entre si através da relação de dependência, como sujeito e verbo, objeto e verbo, dentro outros (Kübler, 2009).

Análise semântica é a etapa que tem por objetivo determinar o significado das palavras e frases, estabelecendo relações semânticas entre elas. Essa etapa é essencial para o entendimento do contexto e do conteúdo das palavras, sendo utilizada para tarefas de processamento de linguagem natural que envolvem tradução e geração automática de textos (Jurafsky, 2000). Além disso, essa técnica tem um grande potencial para a comunidade científica no sentido de melhorar as ontologias existentes na “Web” e os sistemas de representação de conhecimentos (Goddard, 2011). Modelos formais de representação de conhecimento, conhecidos como ontologias, descrevem as relações entre diferentes conceitos e entidades. São amplamente utilizadas para analisar o significado de palavras e

frases em um texto. Elas são projetadas para serem processadas por computadores e podem ser utilizadas em tarefas como extração de informação, classificação de documentos, geração de perguntas-respostas e tradução automática (Felsen, 2002).

Já a análise pragmática é uma técnica da linguística que se concentra em entender o significado das palavras dentro de um contexto específico. Essa técnica consegue apurar aspectos como implícito, inferência e como a linguagem é utilizada em diferentes situações sociais e culturais. A compreensão da pragmática é fundamental para se entender como a linguagem funciona na comunicação humana e é amplamente estudada em áreas como a linguística, psicologia e outras disciplinas relacionadas (Vicente Cruz et al., 1996).

Resumidamente, os estágios analíticos do processamento de linguagem natural são essenciais para que a máquina possa entender e processar dados textuais da melhor forma possível.

A PLN tem sua importância nos seguimentos científicos, econômicos, sociais e culturais. O ponto positivo é que essa tecnologia está em constante crescimento devido à implementação de suas técnicas em uma variedade de novas aplicações de linguagem, como descrito anteriormente. Por esta razão, é importante para uma ampla gama de pessoas, ter competências práticas para a utilização dessas técnicas englobadas pela PLN como, por exemplo, analistas de dados, engenheiro de dados, cientista de dados, desenvolvedores de “softwares”, cientistas da computação, acadêmicos, dentre outros (Bird et al., 2009).

NLTK é uma plataforma que trabalha com as diversas técnicas de processamento estatístico de linguagem natural, facilitando a implementação dessas ferramentas em diversos “softwares”. A plataforma possui recursos léxicos e disponibiliza bibliotecas de processamento de textos para tokenização, classificação, “stemming”, análise de raciocínio semântico, e um fórum ativo de discussão (Bird et al., 2009). Para desenvolver aplicações utilizando a biblioteca de código aberto NLTK, é necessário a utilização da linguagem de programação python[2].

Python é uma linguagem de alto nível, orientada a objetos, interpretada e iterativa, de tipagem forte e dinâmica, com uma sintaxe relativamente simples e de fácil compreensão. A linguagem possui diversas estruturas como, por exemplo, dicionários, “arrays”, e uma gama de bibliotecas e módulos prontos para atender grandes demandas de processamento (Borges, 2014).

Uma das bibliotecas mais populares do Python é a NumPy, que manipula matrizes multidimensionais e realiza operações matemáticas de alto desempenho (Pedregosa et al., 2011). Pandas também é uma biblioteca muito importante para realizar tarefas de limpeza e preparação dos dados, pois oferece ao usuário ferramentas analíticas robustas para trabalhar

[2] Página oficial: https://www.python.org/

com dados estruturados e fornece estrutura de dados de alto desempenho, tonando-se fundamental para a análise exploratória dos dados (McKinney, 2010).

De forma simplificada, bibliotecas em Python são um conjunto de ferramentas extremamente poderosas disponibilizadas ao usuário, no qual se torna possível realizar uma variedade de tarefas de forma eficiente (Borges, 2014). Também são fundamentais para o reaproveitamento de código.

No entanto, para ter acesso a todas as ferramentas de análise de dados disponibilizadas pela linguagem, é necessário instalar algumas dependências do python em um "hardware" com poder de processamento consideravelmente robusto. Visto que isso pode se tornar um limitante, existem algumas ferramentas de computação em nuvem que disponibiliza uma máquina virtual para realizar essa atividade, como por exemplo o Google Colab[3].

O Google Colab é uma plataforma de computação em nuvem gratuita, baseada em Jupyter Notebooks, que permite aos usuários escrever e executar código em Python. Ele é mantido pelo Google Research e é uma ótima opção para quem procura uma plataforma de desenvolvimento colaborativo e acessível (Gunawan, 2020). A vantagem de usar uma plataforma de como o Google Colab é que ela permite acesso a recursos computacionais de alto desempenho, como Unidades de Processamento Gráfico "Graphics Processing Unit" [GPU] e Unidades de Processamento Tensorial "Tensor Processing Unit" [TPU], sem a necessidade de configurar e gerenciar infraestrutura própria. Isso torna o Colab especialmente útil para aplicações de aprendizado de máquina e processamento de grandes conjuntos de dados (Carneiro, 2018).

A extração de dados utilizados para exemplificar o funcionamento da biblioteca NLTK refere-se a arquivos textuais em formato txt. Esse formato foi escolhido por ser um documento de estrutura simples e de fácil manipulação, contendo sequências de linhas com descrições de caracteres. Outro ponto positivo de arquivos com extensões .txt é a sua compatibilidade com todos os sistemas operacionais como Linux, Mac, Windows, Android e iOS, por exemplo.

Após a etapa de extração dos dados, é essencial realizar a tarefa de tratar as informações coletadas, visando retirar itens indesejados que possam interferir nas análises futuras. Para tratar essa situação, a implementação de técnicas de expressão regular, ou "Regex"[4], são frequentemente utilizadas, pois permitem identificar sequências de caracteres ou determinadas combinações considerando os parâmetros definidos pelo usuário. Pode-se obter resultados significativos utilizando essa ferramenta como, por exemplo, separar letras de números, identificar documentos seguindo padrões de caracteres (CPF, por exemplo),

[3] Disponível em: https://colab.research.google.com
[4] Termo abreviado do inglês "regular expression".

identificar extensões de documentos, dentre outros parâmetros. De forma simplificada, o desenvolvimento de funções com o objetivo de realizar o pré-processamento dos dados textuais contidos nos arquivos, é de grande importância para a parte analítica do projeto.

Com os dados obtidos após a implementação dos processos citados anteriormente, optou-se por desenvolver uma análise qualitativa das informações, onde o intuito é explorar e compreender as principais funcionalidades da biblioteca NLTK.

Resultados e Discussões

Neste estágio, será realizada uma avaliação inicial dos dados coletados e dos métodos utilizados para analisá-los. O objetivo é fornecer uma compreensão geral dos resultados obtidos e identificar quaisquer problemas ou limitações que possam afetar o objetivo final do projeto. Para que isso seja possível, é necessário descrever detalhadamente as principais técnicas utilizadas para o processamento de dados textuais usando a biblioteca NLTK e linguagem de programação “python”, exemplificando alguns cenários onde essas técnicas podem ser aplicadas para extrair uma base de informações consistentes e significantes.

O desenvolvimento de um algoritmo de processamento de linguagem natural foi realizado seguindo etapas lineares para alcançar o objetivo final do projeto, que é, de forma resumida, gerar “insights” com informações extraídas de arquivos de texto. Nesse sentido, cada etapa do desenvolvimento da análise de dados textuais foi separada em blocos, para facilitar o entendimento de cada “script”.

Nesse caso, para iniciar a etapa de utilização das funções do processamento de linguagem natural, é necessário importar a biblioteca NLTK para dentro do projeto. A Figura 3 exemplifica como funciona o processo de importação de bibliotecas utilizando a linguagem de programação “python” e a plataforma de computação em nuvem Google Colab.

```
1 import nltk
```

Figura 3. Importação da biblioteca NLTK para dentro do projeto
Fonte: Resultados originais da pesquisa

Com a importação da biblioteca feita, é necessário realizar o “download” dos módulos contidos dentro da biblioteca NLTK. Essa tarefa é extremamente importante para o desenvolvimento do projeto, pois esses módulos que serão instanciados para dentro do projeto ficará acessível em toda a extensão do “script”, sendo possível acionar quaisquer funções a qualquer momento, dependendo da necessidade do usuário. A Figura 4 mostra como realizar o download desses módulos.

```
nltk.download('all')
```

Figura 4. “Download” de todos os módulos contidos na biblioteca NLTK
Fonte: Resultados originais da pesquisa

Conforme informado anteriormente, a biblioteca NLTK disponibiliza diversos módulos de trabalho, que se tornam acessíveis após a etapa de ”download”. Um módulo bastante utilizado para obtenção de dados textuais é o “corpus”.

“Corpus” é um conjunto de textos utilizados para treinar e testar algoritmos de processamento de linguagem natural. A biblioteca NLTK possui uma série de “corpus” pré-carregados, que podem ser utilizados para diversas tarefas, como análise de sentimento, classificação de texto, entre outras (Bird et al., 2009). Para complementar, o autor informa que literatura clássica, diálogos, notícias e até jargão técnico, são exemplos de variedades de “corpus” existentes na biblioteca NLTK, assim como o gutenberg, que contém milhares de livros clássicos, e o webtext, que contém textos de sites da web. Esses conjuntos fornecem uma base robusta de dados textuais que serão utilizados para treinar e testar algoritmos de processamento de linguagem natural (Bird et al., 2009).

A Figura 5 demonstra na prática como funciona para ter acesso ao conjunto de dados Gutemberg da biblioteca NLTK. A função “fileids()” consegue listar todas as obras disponibilizadas dentro do “corpus” especificado.

```
for i in nltk.corpus.gutenberg.fileids():
  print(i)

>>> austen-emma.txt
>>> austen-persuasion.txt
>>> austen-sense.txt
>>> bible-kjv.txt
>>> blake-poems.txt
>>> bryant-stories.txt
>>> burgess-busterbrown.txt
```

Figura 5. Verificação dos arquivos textuais contidos no corpus Gutenberg
Fonte: Resultados originais da pesquisa

A biblioteca NLTK disponibiliza alguns conteúdos em português, como obras do escritor Joaquim Maria Machado de Assis. Executando o trecho de código apresentado na

Figura 6, é possível visualizar algumas das obras disponibilizadas publicamente[5] pela biblioteca. A imagem foi adaptada devido a quantidade de obras contidas na biblioteca.

```
print(nltk.corpus.machado.readme())

>>> romance/marm01.txt: Ressurreição (1872)

>>> contos/macn001.txt: Contos Fluminenses (1870); Miss Dollar; Luís
↪  Soares; A mulher de preto; O segredo de Augusta; Confissões de uma
↪  viúva moça; Linha reta e linha curva; Frei Sim
>>> contos/macn002.txt: Histórias da meia-noite (1873); A parasita
↪  azul; As bodas de Luís Duarte; Ernesto de Tal; Aurora sem dia; O
↪  relógio de ouro; Ponto de vista

>>> cronica/macr01.txt: Comentários da semana (1861-1863)

>>> critica/mact21.txt: Propósito (1866)
```

Figura 6. Identificação das obras contidas no corpus Machado da biblioteca NLTK
Fonte: Resultados originais da pesquisa

Com a visualização da listagem de obras disponibilizada pela biblioteca NLTK, é possível notar algumas coisas, como por exemplo a categoria – romance, poesia, crítica, dentre outras –, um código e, em seguida, o título da obra. Essa distinção se faz necessária para facilitar o acesso a cada um dos conjuntos textuais. A Figura 7 exemplifica como funciona a etapa de acessar o conjunto e visualizar o texto contido nele.

```
_conjuntotxt = nltk.corpus.machado.raw('critica/mact15.txt')

print(_conjuntotxt[963:1317])

>>> Deduzir de semelhante estado a culpa do público, seria transformar
↪  o efeito em causa. O público não tem culpa nenhuma, nem do estado
↪  da arte, nem da sua indiferença por ela; uma prova disso é a
↪  solicitude com que corre a ver a primeira representação das peças
↪  nacionais, e os aplausos com que sempre recebe os autores e as
↪  obras, ainda as  menos corretas.
```

Figura 7. Visualização das informações contidas no conjunto de dados textuais
Fonte: Resultados originais da pesquisa

[5] As obras citadas estão disponibilizadas no domínio público: http://machado.mec.gov.br/

No entanto, devido a diversidade proporcionada pela biblioteca NLTK, é possível utilizar todas as funções em uma base de dados textuais externa. Basicamente, é possível realizar a leitura das informações contidas em um arquivo .txt, por exemplo, e utilizar a parte analítica da biblioteca para obter “insights” desses dados. O funcionamento de leitura de arquivos .txt por uma linguagem de programação é linear, iniciando pela abertura do arquivo em si e, em seguida, realizando a leitura das informações contidas no arquivo, linha por linha. Após esse processo, é necessário armazenar as informações em um estrutura de dados apropriada, como uma lista ou dicionário.

Estruturas de dados são tipos especiais de dados que permitem armazenar e manipular informações de maneira eficiente. Estruturas de dados chamadas Listas são ordenadas e podem ser modificadas. Elas permitem guardar uma série de itens, que podem ser de vários tipos de dados, até mesmo outras listas. Essas estruturas são criadas usando colchetes [] e separando seus elementos por vírgulas. Já as estruturas de dados conhecidas como Dicionários são desordenadas e permitem vincular chaves e valores. Cada chave é exclusiva e seu correspondente valor pode ser de diferentes tipos. Eles são criados usando chaves {} e separando as chaves e valores com dois pontos. Uma tupla é uma coleção imutável de elementos que podem ser de tipos diferentes, e são criadas usando os parênteses (). Essas informações podem ser encontradas no site oficial da linguagem “python”[6]. A imagem a seguir mostra esses tipos de estrutura na prática.

```
1  lista = [1, 2, 3, 4]
2  dicionario = {'nome': 'Vinicius', 'idade': 29, 'cpf': 12345678910}
3  tupla = (1, 2, 3, 4)
```

Figura 8. Exemplos de estruturas de dados utilizando a linguagem de programação Python
Fonte: Resultados originais da pesquisa

Sendo assim, seguindo a codificação do script que realiza o processamento de linguagem natural, a próxima etapa contém o caminho onde será possível acessar os arquivos textuais que vão ser lidos e analisados. Como a plataforma Google Colab está sendo utilizada para o desenvolvimento deste projeto, a integração com outra ferramenta de armazenamento de arquivos em nuvem, da mesma empresa, se faz mais adequada para o proposito descrito. Essa ferramenta é o Google Drive[7].

Com o Google Drive importado para o projeto, a próxima tarefa seria realizar a leitura de um arquivo .txt que se encontra dentro de algum caminho. Sendo assim, os arquivos

[6] Python tutorial: https://docs.python.org/3/tutorial/index.html
[7] Disponível em: https://www.google.com/intl/pt-BR/drive/

escolhidos para serem analisados são os livros da saga Harry Potter, elaborados pela escritora Joanne Kathleen Rowling[8], popularmente conhecida como J. K. Rowling.

```
from google.colab import drive
drive.mount('/content/drive')

import os
_path_books_drive = '/content/drive/MyDrive/Harry_Potter/'
list_books_hp = sorted(os.listdir(_path_books_drive))
list_books_hp

>>>["Book 1 - The Philosopher's Stone.txt",
 'Book 2 - The Chamber of Secrets.txt',
 'Book 3 - The Prisoner of Azkaban.txt',
 'Book 4 - The Goblet of Fire.txt',
 'Book 5 - The Order of the Phoenix.txt',
 'Book 6 - The Half Blood Prince.txt',
 'Book 7 - The Deathly Hallows.txt']
```

Figura 9. Leitura de arquivos .txt armazenados no Google Drive
Fonte: Resultados originais da pesquisa

Conforme demonstrado na Figura 9, é possível visualizar como funciona o processo de identificação dos arquivos em uma pasta alocada no Google Drive. Algumas bibliotecas específicas da linguagem “python” tiveram que ser importadas para auxiliar nesse processo, como por exemplo a importação da biblioteca “os”, que fornece funcionalidades relacionadas ao sistema operacional, permitindo que o usuário realize manipulações de diretórios e arquivos, interação com outros sistemas de arquivos, acesso a algumas informações do sistema operacional, dentre outras atividades. No caso acima, foi utilizado a funcionalidade de navegar entre diretórios e listar quais arquivos estavam contidos na pasta em si. O retorno dessa solicitação foi uma estrutura de dados do tipo lista, contendo os nomes de todos os livros da saga Harry Potter.

Com os arquivos instanciados no projeto, é possível “printar” uma cadeia de caracteres e visualizar como os dados textuais estão estruturados nesse arquivo. A Figura 10 mostra exatamente essa questão.

[8] Bibliografia disponível em: https://www.jkrowling.com/about/

```
books_hp_no_processed[0]

>>> '/ \n\n\n\n\nTHE BOY WHO LIVED \n\nMr. and Mrs. Dursley, of number
↪  four, Privet Drive, \nwere proud to say that they were perfectly
↪  normal...'
```

Figura 10. “Print” de uma parte do livro Harry Potter e a Pedra Filosofal sem nenhuma aplicação de pré-processamento
Fonte: Resultados originais da pesquisa

Como pode ser visto na Figura 10, os dados textuais dentro do arquivo estão com diversos caracteres que podem interferir na parte analítica do projeto. Nesse caso, para remover essas informações, o ideal é fazer um pré-processamento dos dados realizando todas as limpezas necessárias dessas informações irrelevantes.

A tratativa de pré-processamento é executada de acordo com a estrutura de dados que cada projeto está relacionado e, para cada caso, uma ação deve ser tomada. A tratativa ideal para alcançar os objetivos deste projeto foi: executar uma função interna da linguagem “python” para deixar todas as letras minúsculas, remoção de qualquer caractere diferente de uma letra utilizando técnicas de expressão regular, e remoção de dados específicos do arquivo do livro como, por exemplo, o rodapé, que continha descrições de páginas, nome do livro e nomenclaturas autorais e editoriais. Adjunto, foi implementada uma função que remove todas as “stopwords” do texto.

De acordo com Silge (2017), “stopwords” são termos comuns, como "e", "de", "um", "o", dentre outros, que são desconsiderados em análises de linguagem natural porque não possuem um significado relevante. Essas palavras geralmente são removidas antes de realizar análises, pois a sua presença pode prejudicar resultados e usar recursos computacionais de forma desnecessária. A lista de “stopwords” pode ser diferente dependendo do idioma, objetivo e da aplicação da análise. O uso de “stopwords” é uma técnica importante para aprimorar a eficiência e precisão de muitos processos de análise de linguagem natural.

A Figura 11 demonstra como ficaram os dados textuais após a aplicação do pré-processamento.

```
books_hp_processed[0]

>>> 'boy lived mr mrs dursley number four privet drive proud say
↪  perfectly normal...'
```

Figura 11. “Print” de uma parte do livro Harry Potter e a Pedra Filosofal com a aplicação do pré-processamento
Fonte: Resultados originais da pesquisa

Essa tratativa de remover palavras que não tem significância para o contexto auxilia na etapa de visualização das informações textuais do arquivo que foi analisado. Para exemplificar essa visualização de dados, foi aplicado a função “FreqDist” no livro Harry Potter e a Pedra Filosofal. “FreqDist” é uma ferramenta disponibilizada pela biblioteca NLTK que possibilita registrar a frequência que cada palavra aparece dentro de um conjunto de caracteres. A Figura 12 representa graficamente a frequência de cada palavra.

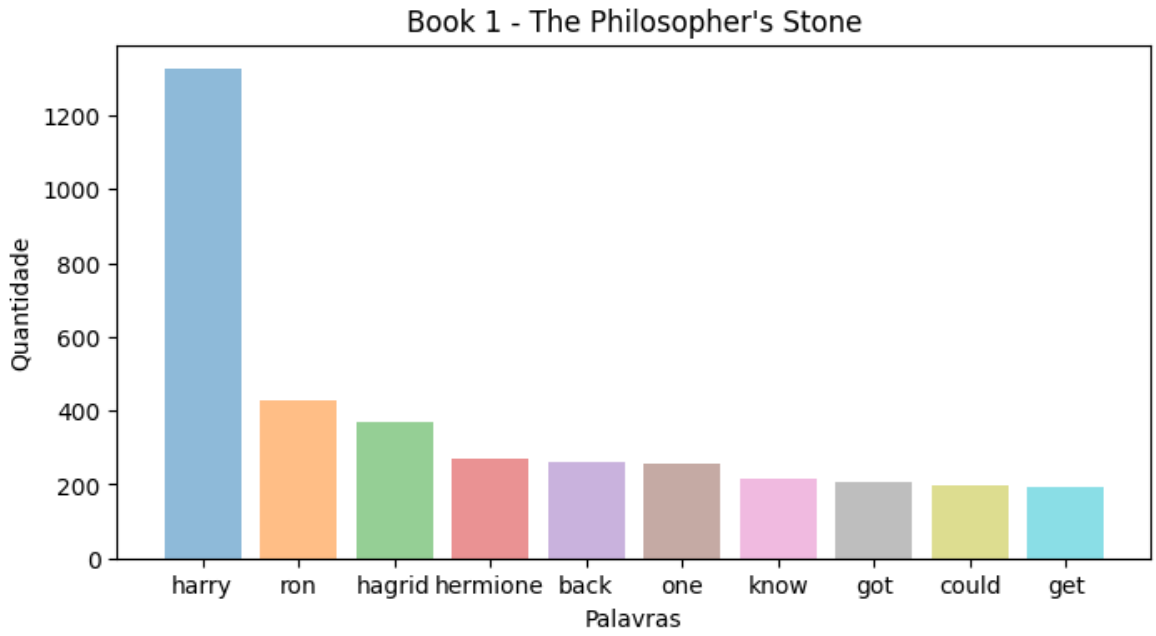

Figura 12. Palavras que mais se repete no livro Harry Potter e a Pedra Filosofal com a aplicação do pré-processamento
Fonte: Resultados originais da pesquisa

Dessa forma, é possível interpretar facilmente quais informações estão contidas no gráfico. No entanto, a Figura 13 demonstra como ficaria uma representação gráfica em caso de não realização da etapa de pré-processamento.

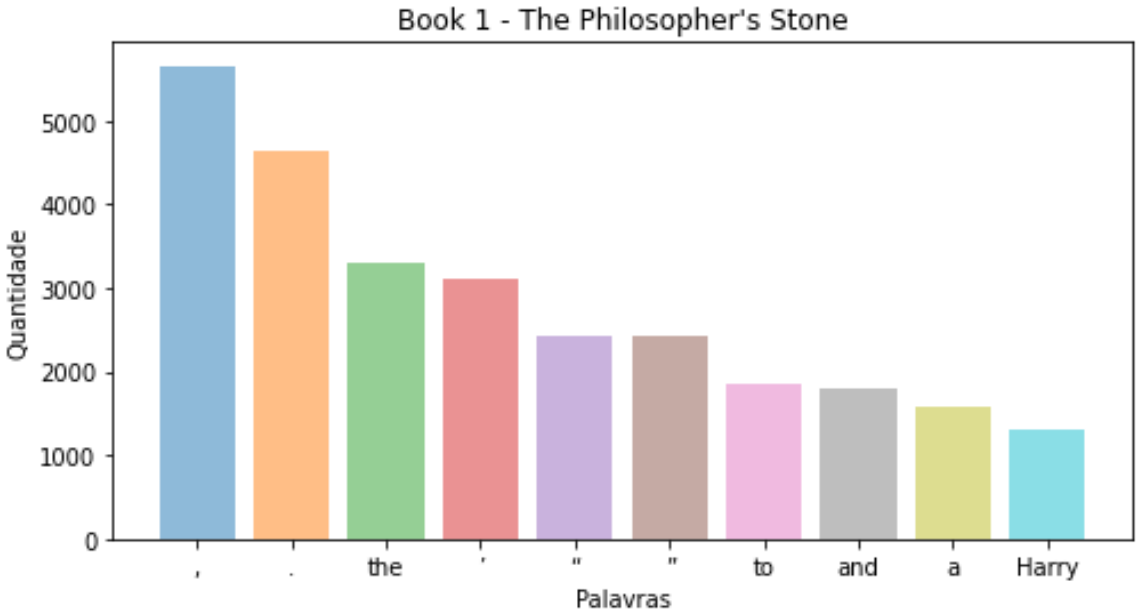

Figura 13. Palavras que mais se repete no livro Harry Potter e a Pedra Filosofal sem a aplicação do pré-processamento
Fonte: Resultados originais da pesquisa

As exemplificações gráficas acima são referentes ao “output” gerado pelo processo e demonstrado pela Figura 14. Essa etapa é importante para definir qual tipo de processamento terá eficiência na base de dados que será manipulada.

```
print(prob_hp_noprocess_philosopher_stone.most_common(5))
print(prob_hp_philosopher_stone.most_common(5))

>>>[(',', 5658), ('.', 4639), ('the', 3312), ('’', 3111), ('"', 2437)]

>>>[('harry', 1325), ('said', 794), ('ron', 429), ('hagrid', 370),
 ↪  ('hermione', 269)]
```

Figura 14. Output gerado após o processamento textual do livro Harry Potter e a Pedra Filosofal, limitado a cinco palavras que mais se repete, onde é possível representar a diferença das informações não tratadas e a aplicação do pré-processamento.
Fonte: Resultados originais da pesquisa

A biblioteca NLTK disponibiliza outra ferramenta interessante para a parte de análise textual das informações, chamada similaridade. Para a exemplificação dessa técnica, é necessário informar que será utilizado a base de dados formatada, ou seja, informações processadas e tratadas.

Dentro do tema processamento de linguagem natural, a similaridade pode ser definida como uma medida que avalia o grau de semelhança entre dois ou mais elementos linguísticos.

Essa ferramenta é bastante utilizada para resolver trabalhos como recuperação de informações, categorização e classificação de sentimentos (Jurafsky, 2000).

Mensurar a similaridade de uma palavra em uma base de dados textual requer a aplicação de técnicas como a análise de similaridade baseada em vetorização de palavras deve ser aplicada. Outras técnicas como análise semântica latente [LSA] e distribuída [DSA], que utilizam cálculos estatísticos e espaço vetorial para mensurar a similaridade das palavras analisadas (Mikolov, 2013).

Para aplicar a função de similaridade disponibilizada pela biblioteca NLTK, foi utilizado o segundo livro da saga Harry Potter, chamado “The Chamber of Secrets”. Após coletar os dados textuais desse livro e aplicar todas as técnicas de pré-processamento citadas anteriormente no projeto, a implementação da função de similaridade foi aplicada com a finalidade de encontrar correlações com a palavra “basilisk”[9].

```
1  analytics_hp_chamber_secrets.similar('basilisk')
2
3  >>>said snake serpent
```

Figura 15. Output gerado após a aplicação da técnica de similaridade no livro Harry Potter e a Câmara Secreta, com os dados textuais pré-processados.
Fonte: Resultados originais da pesquisa

Como por ser visto na Figura 15, a busca de similaridade pela palavra “basilisk” gerou os seguintes “output”: “said”, “snake”, “serpent”. Com essas informações, já existe uma possibilidade de deduzir o que seria um “basilisk”.

No entanto, após implementar a busca de similaridade entre as palavras, a biblioteca NLTK também oferece uma função chamada “concordance”. A concordância é uma ferramenta importante na análise linguística e no processamento de linguagem natural, pois permite examinar como as palavras são usadas em diferentes contextos e, em particular, como elas concordam com outras palavras em uma frase (Sinclair, 1991).

Utilizando também a base de dados formatada, foi realizado uma pesquisa de concordância com a mesma palavra do exemplo anterior, “basilisk”. A Figura 16 demonstra essa aplicação.

[9] No universo de Harry Potter, “Basilisk” é uma serpente que pode crescer imensamente. É uma criatura mágica e rara, também conhecida como o “King of Serpents”. Mais detalhes: https://harrypotter.fandom.com/wiki/Basilisk

```
analytics_hp_chamber_secrets.concordance('basilisk')

>>>deadly basilisk known also king serpents snake

>>>deadly venomous fangs basilisk murderous stare fixed beam eye

>>>suffer instant death spiders flee basilisk mortal enemy basilisk

>>>monster chamber basilisk giant serpent hearing voice
```

Figura 16. Output gerado após a aplicação da técnica de concordância no livro Harry Potter e a Câmara Secreta, com os dados textuais pré-processados.
Fonte: Resultados originais da pesquisa

Após a implementação das funções de similaridade e concordância utilizando o mesmo parâmetro de pesquisa, deduzir com maior assertividade o que seria um “basilisk” fica mais fácil. Mesmo para uma pessoa que não conhece a saga Harry Potter, com o processamento de linguagem natural aplicado nas sentenças anteriores, é possível assimilar que “basilisk” é um tipo de serpente.

Para complementar, a técnica de análise textual denominada bigrama – também conhecido como pares de palavras consecutivas – também está implementada na biblioteca NLTK. Basicamente, um bigrama é uma sequência de duas palavras consecutivas em um texto. A utilização dessa técnica pode ser útil para identificar padrões de uso de palavras em bases de dados textuais e para extrair informações relevantes para uma variedade de aplicações, como reconhecimento de fala, tradução automática, análise de sentimento e outras tarefas de processamento de linguagem natural (Manning, 1999).

```
analytics_hp_chamber_secrets.collocations()

>>>professor mcgonagall; uncle vernon; mrs weasley; chamber secrets;
↪  fred george; headless nick; madam pomfrey; nearly headless; harry
↪  potter; gilderoy lockhart; moaning myrtle; aunt petunia; hospital
↪  wing; mrs norris; common room; sorting hat; professor sprout;
↪  crabbe goyle; ron hermione; great hall
```

Figura 17. Output gerado após a aplicação da técnica de bigramas no livro Harry Potter e a Câmara Secreta, com os dados textuais pré-processados.
Fonte: Resultados originais da pesquisa

Após a implementação das técnicas descritas na base de dados relacionada aos livros, foi adicionado “datasets” contento os diálogos dos personagens que foram adaptados para os filmes da saga. Sendo assim, as mesmas técnicas de pré-processamento foram aplicadas

nos dados textuais dos filmes com o intuído de fazer uma comparação das frequências de palavras que tem em cada “dataset”.

As Tabelas 1 e 2 representam um comparativo entre os dados textuais demonstrados pelos diálogos dos filmes e os dados textuais que foram extraídos dos livros. Ambas as tabelas exemplificam o impacto da etapa de pré-processamento para normalização das bases de dados.

Tabela 1. Quantidade de palavras em cada fonte de dados, com a aplicação da etapa de pré-processamento, e percentual de redução na frequência de palavras dos filmes se comparado com os livros.

Nome do livro/filme	Filme	Livro	Porcentagem
The Philosopher's Stone	4801	39907	12,03%
The Chamber of Secrets	5429	45096	12,04%
The Prisoner of Azkaban	4752	56366	8,43%
The Goblet of Fire	3943	98844	3,89%
The Order of the Phoenix	4595	132023	3,48%
The Half Blood Prince	5383	85961	6,23%
The Deathly Hallows	7773	100537	7,73%

Fonte: Resultados originais da pesquisa

Tabela 2. Quantidade de palavras em cada fonte de dados, sem a aplicação da etapa de pré-processamento, e percentual de redução na frequência de palavras dos filmes se comparado com os livros.

Nome do livro/filme	Filme	Livro	Porcentagem
The Philosopher's Stone	12273	101227	12,12%
The Chamber of Secrets	13243	111538	11,87%
The Prisoner of Azkaban	12027	142306	8,45%
The Goblet of Fire	9360	247523	3,78%
The Order of the Phoenix	11501	332518	3,46%
The Half Blood Prince	13849	220235	6,29%
The Deathly Hallows	20186	255569	7,90%

Fonte: Resultados originais da pesquisa

Com esse resultado, é possível notar que as frequências de palavras que foram expressas durante o filme são menores que as descritas nos livros. Porém, essa queda é justificada. De acordo com Desmond e Hawkes (2006), a adaptação cinematográfica de um livro é um processo complexo que envolve a seleção e transformação de elementos da obra original para o formato audiovisual. Uma das principais tarefas da adaptação de livros para filmes está relacionada ao entendimento central da história, o que pode acarretar a eliminação

de cenas ou personagens inteiros e, consequentemente, na redução da quantidade de palavras utilizadas.

De forma resumida, os livros precisam descrever detalhadamente um local, um personagem, uma ação ou reação de uma cena específica, animais místicos, funcionamento de uma magia e diversas outras ações que ocorrem no decorrer da história. Já no filmes, esses cenários são substituídos por imagens, não sendo necessário descrever, de forma textual, suas características. Outra razão que impacta diretamente na redução da frequência de palavras está relacionada os diálogos nos filmes de Harry Potter, que são mais sucinto do que nos livros (Burkhardt e Gauvain, 2013).

Outra variável a ser mensurada é a proporção de “stopwords” e caracteres especiais que foram removidos em ambas as fontes de dados textuais. As Tabelas 3 e 4 representam detalhadamente essa proporção em todos os livros e filmes, bem como a média total que foi removida.

Tabela 3. Percentual de “stopwords” e caracteres especiais removidos na base de dados relacionadas aos filmes, após a etapa de pré-processamento dos dados.

Nome do filme	Texto Original	Pré-processado	Porcentagem
The Philosopher's Stone	4801	12273	38,12%
The Chamber of Secrets	5429	13243	41,00%
The Prisoner of Azkaban	4752	12027	39,51%
The Goblet of Fire	3943	9360	42,13%
The Order of the Phoenix	4595	11501	39,95%
The Half Blood Prince	5383	13849	38,87%
The Deathly Hallows	7773	20186	38,51%
TOTAL:	36676	92439	39,68%

Fonte: Resultados originais da pesquisa

Tabela 4. Percentual de “stopwords” e caracteres especiais removidos na base de dados relacionadas aos livros, após a etapa de pré-processamento dos dados.

Nome do livro	Texto Original	Pré-processado	Porcentagem
The Philosopher's Stone	39907	101227	39,42%
The Chamber of Secrets	45096	111538	40,43%
The Prisoner of Azkaban	56366	142306	39,61%
The Goblet of Fire	98844	247523	39,93%
The Order of the Phoenix	132023	332518	39,70%
The Half Blood Prince	85961	220235	39,03%
The Deathly Hallows	100537	255569	39,34%
TOTAL:	558734	1410916	39,60%

Fonte: Resultados originais da pesquisa

Como pode ser observado, a remoção foi equilibrada em ambos os lados, demonstrando que o entendimento da base de dados que foi implementado durante a análise dos livros pode ser facilmente replicado na análise dos diálogos nos filmes.

De forma resumida, foi constatado que, em média, a frequência de palavras nos filmes da saga Harry Potter apresenta uma redução de 7,69% em comparação aos livros. Esse resultado sugere que a adaptação dos livros para o formato de filmes pode ter um impacto na quantidade de palavras utilizadas na narrativa, reforçando a afirmação de Desmond e Hawkes (2006) citada anteriormente.

Conclusão(ões) ou Considerações Finais

Em resumo, o presente trabalho teve como objetivo analisar os livros da série Harry Potter sob a perspectiva do processamento de dados textuais. Foi possível verificar a importância da utilização de técnicas computacionais para análise de grandes volumes de dados textuais, permitindo a extração de informações relevantes. Ademais, foi possível tirar algumas conclusões sobre a implementação deste projeto:

- Entender a estrutura/formatação dos dados que serão analisados é de suma importância para a etapa de pré-processamento das informações;
- A etapa de pré-processamento implementada no projeto foi significante para uma análise mais assertiva;
- Um fluxo de extração, transformação e armazenamento dos dados robusto será decisivo para as etapas analíticas seguintes;
- As ferramentas disponibilizadas pela biblioteca NLTK se demonstraram robustas o suficiente para tratar grande volume de dados sem gargalos;
- Foi possível constatar o funcionamento analítico da biblioteca em mais de um idioma, sem perder eficiência;
- Foi possível gerar insumos básicos que podem ser úteis para a classificação dos dados textuais;
- A biblioteca NLTK disponibiliza uma gama de funcionalidades analíticas avançadas, envolvendo cálculos estatísticos e probabilísticos. Porém, para a usabilidade dessas ferramentas, se faz necessário ter certo conhecimento sobre o tema;
- As técnicas de similaridade, concordância e bigramas foram implementadas neste trabalho. Com a aplicação dessas técnicas, mesmo para quem nunca teve nenhum contato com o universo de Harry Potter, foi possível identificar o significado de uma palavra demonstrando sua aplicação dentro de vários contextos.

Com essas informações extraídas durante a elaboração do projeto, também é possível visualizar alguns temas a serem desenvolvidos como propostas futuras, por exemplo:

- Conforme mencionado, a biblioteca NLTK disponibiliza funções probabilísticas e estatísticas. Uma das técnicas de estatística disponibilizadas é a de Good-Turing. Nesse caso, existe a possibilidade de utilizar essas ferramentas para criar modelos que identificam padrões de escrita;
- Implementação do processamento de dados textuais seguindo as categorias disponibilizadas pela biblioteca;
- Desenvolvimento de aplicações “WebScraping” implementadas em conjunto com a biblioteca NLTK é bastante significante. Testes básicos foram desenvolvidos durante a elaboração do projeto e esse conjunto se mostrou bastante significante em alguns temas. Por exemplo, coletar dados sobre uma empresa X e implementar a PLN para extrair informações subjetivas expressas em textos, como opiniões, emoções, atitudes e sentimentos de indivíduos ou grupos. De forma resumida, é possível desenvolver uma análise de sentimento sobre empresas de diversos seguimentos para descobrir informações, como padrões de clientes, opiniões sobre os produtos lançados, “insights” para criação de um novo produto, estilos/modas do momento, dentre outros.

Agradecimentos

Gostaria de agradecer a Deus por me sustentar durante todo o processo de elaboração deste trabalho. Agradeço também à minha esposa pelo apoio incondicional e incentivo constante. Também agradeço ao meu orientador que teve um papel importante para a excelência deste projeto.

Referências

Bird, Steven, Ewan Klein, and Edward Loper. Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc.", 2009.

Borges, Luiz Eduardo. Python para desenvolvedores: aborda Python 3.3. Novatec Editora, 2014.

Burkhardt, J. M., & Gauvain, M. C. (2013). Comparing the Books to the Movies: A Study of Harry Potter and the Philosopher's Stone. Children's Literature in Education, 44(3), 257-273.

Carneiro, Tiago et al. Performance analysis of google colaboratory as a tool for accelerating deep learning applications. IEEE Access, v. 6, p. 61677-61685, 2018.

Castro, Sergio M. et al. Automated annotation and classification of BI-RADS assessment from radiology reports. Journal of biomedical informatics, v. 69, p. 177-187, 2017.

Chang, Angel X.; Manning, Christopher D. TokensRegex: Defining cascaded regular expressions over tokens. Stanford University Computer Science Technical Reports. CSTR, v. 2, p. 2014, 2014.

Charniak, Eugene. Statistical parsing with a context-free grammar and word statistics. AAAI/IAAI, v. 2005, n. 598-603, p. 18, 1997.

Chomsky, Noam. Syntactic structures. In: Syntactic Structures. De Gruyter Mouton, 2009.

Dale, Robert; MOISL, Hermann; SOMERS, Harold (Ed.). Handbook of natural language processing. CRC press, 2000.

Desmond, John, and Peter Hawkes. Adaptation: Studying film and literature. McGraw-Hill Humanities Social, 2006.

Felsen, Dieter et al. Semantic web application areas. In: NLDB Workshop. sn, 2002.

Gazeau, Valentin, and Cihan Varol. "Automatic spoken language recognition with neural networks." Int. J. Inf. Technol. Comput. Sci.(IJITCS) 10.8 (2018): 11-17.

Goddard, Cliff. Semantic analysis: A practical introduction. Oxford University Press, 2011.

Gunawan, Teddy Surya et al. Development of video-based emotion recognition using deep learning with Google Colab. TELKOMNIKA (Telecommunication Computing Electronics and Control), v. 18, n. 5, p. 2463-2471, 2020.

Indurkhya, Nitin, and Fred J. Damerau. Handbook of natural language processing. Chapman and Hall/CRC, 2010.

Jiang, Zhipeng et al. De-identification of medical records using conditional random fields and long short-term memory networks. Journal of biomedical informatics, v. 75, p. S43-S53, 2017.

Jurafsky, Dan. Speech & language processing. Pearson Education India, 2000.

Jurafsky, Daniel; MARTIN, James H. An introduction to natural language processing, computational linguistics, and speech recognition. 2000.

Kilgarriff, Adam; Grefenstette, Gregory. Introduction to the special issue on the web as corpus. Computational linguistics, v. 29, n. 3, p. 333-347, 2003.

Kübler, Sandra; MCDONALD, Ryan; NIVRE, Joakim. Dependency parsing. Synthesis lectures on human language technologies, v. 1, n. 1, p. 1-127, 2009.

Manning, Christopher; Schutze, Hinrich. Foundations of statistical natural language processing. MIT press, 1999.

McKinney, Wes et al. Data structures for statistical computing in python. In: Proceedings of the 9th Python in Science Conference. 2010. p. 51-56.

Mikolov, Tomas et al. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

Pedregosa, Fabian et al. Scikit-learn: Machine learning in Python. the Journal of machine Learning research, v. 12, p. 2825-2830, 2011.

Robertson, D. W. "A note on the classical origin of" circumstances" in the medieval confessional." Studies in Philology 43.1 (1946): 6-14.

Python Software Foundation. (s.d.). Tutorial de Python [Documentação oficial]. Recuperado em 1º de fevereiro de 2023, de https://docs.python.org/3/tutorial/index.html

Silge, Julia; Robinson, David. Text mining with R: A tidy approach. " O'Reilly Media, Inc.", 2017.

Sinclair, John. Corpus, concordance, collocation. Oxford University Press, USA, 1991.

Vicente Cruz, Begoña et al. Jacob L. Mey (1993)," Pragmatics. An Introduction", Oxford, Blackwell, 357 págs. 1996.

Wickham, Hadley. "Data analysis." ggplot2. Springer, Cham, 2016. 189-201.

Yang, Jin, et al. "Brief introduction of medical database and data mining technology in big data era." Journal of Evidence-Based Medicine 13.1 (2020): 57-69.