Abordagens de Processamento de Linguagem Natural e Aprendizado Profundo para Classificação de Atos Administrativos de Diário Oficial

David Pereira de Araújo

Universidade Federal de Uberlândia
Faculdade de Computação
Programa de Pós-Graduação em Ciência da Computação

Uberlândia
2024

David Pereira de Araújo

Abordagens de Processamento de Linguagem Natural e Aprendizado Profundo para Classificação de Atos Administrativos de Diário Oficial

Dissertação de mestrado apresentada ao Programa de Pós-graduação da Faculdade de Computação da Universidade Federal de Uberlândia como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Ciência da Computação

Orientador: Henrique Coelho Fernandes
Coorientador: Fabíola Souza Fernandes Pereira

Uberlândia
2024

Ficha Catalográfica Online do Sistema de Bibliotecas da UFU
com dados informados pelo(a) próprio(a) autor(a).

A663 Araujo, David Pereira de, 1987-
2024 Abordagens de Processamento de Linguagem Natural e Aprendizado Profundo para Classificação de Atos Administrativos de Diário Oficial [recurso eletrônico] / David Pereira de Araujo. - 2024.

Orientador: Henrique Coelho Fernandes.
Coorientadora: Fabíola Souza Fernandes Pereira.
Dissertação (Mestrado) - Universidade Federal de Uberlândia, Pós-graduação em Ciência da Computação.
Modo de acesso: Internet.
Disponível em: http://doi.org/10.14393/ufu.di.2024.726
Inclui bibliografia.
Inclui ilustrações.

1. Computação. I. Fernandes, Henrique Coelho,1986-, (Orient.). II. Pereira, Fabíola Souza Fernandes,1987-, (Coorient.). III. Universidade Federal de Uberlândia. Pós-graduação em Ciência da Computação. IV. Título.

CDU: 681.3

Bibliotecários responsáveis pela estrutura de acordo com o AACR2:

Gizele Cristine Nunes do Couto - CRB6/2091
Nelson Marcos Ferreira - CRB6/3074

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
Coordenação do Programa de Pós-Graduação em Ciência da Computação
Av. João Naves de Ávila, 2121, Bloco 1A, Sala 243 - Bairro Santa Mônica, Uberlândia-MG, CEP 38400-902
Telefone: (34) 3239-4470 - www.ppgco.facom.ufu.br - cpgfacom@ufu.br

ATA DE DEFESA - PÓS-GRADUAÇÃO

Programa de Pós-Graduação em:	Ciência da Computação				
Defesa de:	Dissertação, 43/2024, PPGCO				
Data:	29 de novembro de 2024	Hora de início:	13:03	Hora de encerramento:	15:47
Matrícula do Discente:	12212CCP004				
Nome do Discente:	David Pereira de Araujo				
Título do Trabalho:	Abordagens de Processamento de Linguagem Natural e Aprendizado Profundo para Classificação de Atos Administrativos de Diário Oficial				
Área de concentração:	Ciência da Computação				
Linha de pesquisa:	Ciência de Dados				
Projeto de Pesquisa de vinculação:	----------				

Reuniu-se por videoconferência, a Banca Examinadora, designada pelo Colegiado do Programa de Pós-graduação em Ciência da Computação, assim composta: Professores Doutores: Fabíola Souza Fernandes Pereira (Coorientadora)- FACOM/UFU, Thiago Henrique Pereira Silva - FACOM/UFU, Glauco Vitor Pedrosa - PPCA/UnB e Henrique Coelho Fernandes - Cranfield University - UK, orientador do candidato.

Os examinadores participaram desde as seguintes localidades: Glauco Vitor Pedrosa - Brasília/DF, Thiago Henrique Pereira Silva - Belo Horizonte/MG, Henrique Coelho Fernandes - Milton Keynes/Inglaterra. A Coorientadora Fabíola Souza Fernandes Pereira participou da cidade de Uberlândia e o aluno participou da cidade de Campina Verde/MG.

Iniciando os trabalhos o presidente da mesa, Prof. Dr. Henrique Coelho Fernandes, apresentou a Comissão Examinadora e o candidato, agradeceu a presença do público, e concedeu ao Discente a palavra para a exposição do seu trabalho. A duração da apresentação da Discente e o tempo de arguição e resposta foram conforme as normas do Programa.

A seguir o senhor presidente concedeu a palavra, pela ordem sucessivamente, aos examinadores, que passaram a arguir ao candidato. Ultimada a arguição, que se desenvolveu dentro dos termos regimentais, a Banca, em sessão secreta, atribuiu o resultado final, considerando o candidato:

Aprovado

Esta defesa faz parte dos requisitos necessários à obtenção do título de Mestre.

O competente diploma será expedido após cumprimento dos demais requisitos, conforme as normas do Programa, a legislação pertinente e a regulamentação interna da UFU.

Nada mais havendo a tratar foram encerrados os trabalhos. Foi lavrada a presente ata que após lida e achada conforme foi assinada pela Banca Examinadora.

Documento assinado eletronicamente por **Glauco Vitor Pedrosa**, **Usuário Externo**, em 02/12/2024, às 15:23, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Fabíola Souza Fernandes Pereira**, **Professor(a) do Magistério Superior**, em 03/12/2024, às 11:14, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Thiago Henrique Pereira Silva**, **Professor(a) do Magistério Superior**, em 06/12/2024, às 09:25, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Henrique Coelho Fernandes**, **Professor(a) do Magistério Superior**, em 06/12/2024, às 15:07, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://www.sei.ufu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **5896823** e o código CRC **628E2D75**.

Referência: Processo nº 23117.080199/2024-15 SEI nº 5896823

Dedico este trabalho aos meus pais, ***Antonia Lopes*** *e* ***Francisco Lopes****, por todo amor, apoio e ensinamentos que sempre me guiaram; à minha família, por estar ao meu lado em cada etapa deste caminho.*

À minha esposa, ***Mayra****, pelo companheirismo, paciência e incentivo constante, e às minhas filhas,* ***Antonella Araújo*** *e* ***Ana Clara Araújo****, que são minha maior inspiração e motivação para seguir em frente e buscar sempre o melhor.*

A todos vocês, minha gratidão eterna.

Agradecimentos

Em primeiro lugar, agradeço a **Deus** pela dádiva da vida, pela força que me deu para continuar essa caminhada, mesmo nos dias mais difíceis, e pela oportunidade de vivenciar este momento especial. Aos meus pais, **Antonia e Francisco**, por todo o amor, apoio e ensinamentos que sempre me sustentaram. À minha esposa, **Mayra**, e às minhas filhas, **Antonella e Ana Clara Araújo**, pelo o apoio incondicional, por toda a paciência, compreensão e amor ao longo deste percurso no mestrado. Foi por vocês e sem vocês, nada disso faria sentido. E aos meus amigos e colegas mais próximos, que de alguma forma ajudaram ou apoiaram durante essa trajetória. Sou profundamente grato ao meu orientador, **Henrique Fernandes**, pela paciência, compreensão, atenção, e também por compartilhar seus conhecimentos, ensinamentos e incentivos, e à minha coorientadora, **Fabíola Pereira**, que contribuiu no desenvolvimento dessa pesquisa e foram pessoas fundamentais na conclusão deste trabalho. Agradeço a todos que, de alguma forma, me apoiaram e ajudaram, oferecendo palavras de incentivo, conselhos e suporte quando mais precisei. Aos meus colegas de curso, especialmente **Cayo Fontana** e **Franklin Neto**, por toda a parceria e amizade ao longo dessa jornada. Meu sincero agradecimento também aos **professores** da FACOM que contribuíram para minha formação e à **Universidade Federal de Uberlândia (UFU)**, por proporcionar essa oportunidade de crescimento acadêmico e pessoal.

A todos vocês, minha eterna gratidão!

"A força de vontade deve ser mais forte do que a habilidade."
(Boxeador. Muhammad Ali)

Resumo

Este trabalho investigou a aplicação do Processamento de Linguagem Natural (PLN) e Aprendizado Profundo (AP) na automação de processos em instituições públicas, visando melhorar a eficiência e a tomada de decisão baseada em dados. A sobrecarga de informações e a transição para processos digitais têm gerado desafios, sendo um dos principais problemas a resistência de servidores públicos ao uso de tecnologias digitais, com a persistência em métodos manuais para tarefas repetitivas. O objetivo desta pesquisa foi realizar a aplicação de modelos que, utiliza técnicas de PLN e AP, capaz de classificar informações relevantes em textos de atos administrativos extraídos de documentos *Portable Document Format* (PDF) publicados em diário oficial. A fundamentação teórica baseou-se em conceitos de Sistemas de Informação (SI) para analisar os desafios e oportunidades envolvidos na integração dessas tecnologias na administração pública. A metodologia empregada incluiu uma revisão bibliográfica, análise exploratória e experimentos. Foram utilizados algoritmos supervisionados de *Deep Learning* (DL) para desenvolver modelos que classificam informações textuais após um processo de coleta e pré-processamento dos dados obtidos em portais públicos. A abordagem da pesquisa é descritiva e quantitativa, permitindo mensurar a eficiência dos modelos desenvolvidos. Os resultados demonstraram que o modelo *Bidirecional Encoder Representations from Transformers* (BERT) alcançou uma acurácia de 99%, superando modelos descritos na literatura, e se mostrou eficaz na extração e classificação de informações relevantes. Conclui-se que a aplicação de tecnologias de PLN e AP contribui significativamente para a automação de processos e melhoria na tomada de decisões, apresentando um impacto positivo tanto na administração pública quanto na comunidade acadêmica. O trabalho reforça a importância de adotar tecnologias digitais para aprimorar a eficiência e qualidade dos serviços públicos.

Palavras-chave: Processamento de linguagem natural. Deep learning. BERT. Classificação de texto.

Abstract

This study investigated the application of Natural Language Processing (NLP) and Deep Learning (DL) in the automation of processes within public institutions, aiming to improve efficiency and data-driven decision-making. Information overload and the shift toward digital processes have posed challenges, among them the resistance of public officials to adopting digital technologies, as they continue relying on manual methods for repetitive tasks. The objective of this research was to implement models employing NLP and DL techniques capable of classifying relevant information found in administrative acts extracted from Portable Document Format (PDF) documents published in official gazettes. The theoretical framework drew upon Information Systems (IS) concepts to examine the challenges and opportunities in integrating these technologies into public administration. The methodology included a literature review, exploratory analysis, and experimental procedures. Supervised Deep Learning algorithms were used to develop models that classify textual information after a data collection and preprocessing phase from public portals. This descriptive and quantitative research approach enabled an assessment of the developed models' efficiency. The results showed that the Bidirectional Encoder Representations from Transformers (BERT) model achieved a 99% accuracy rate, outperforming models previously described in the literature, and proving effective for extracting and classifying relevant information. The conclusion is that the application of NLP and DL technologies significantly contributes to process automation and improved decision-making, exerting a positive impact on both public administration and the academic community. This work emphasizes the importance of adopting digital technologies to enhance the efficiency and quality of public services.

Keywords: Natural language processing. Deep learning. BERT. Text classification.

Lista de ilustrações

Lista de tabelas

Lista de siglas

AP Aprendizado Profundo

API *Application Programming Interface*

BERT *Bidirecional Encoder Representations from Transformers*

BRNN *Bidirectional Recurrent Neural Networks*

BoW *Bag of Words*

CSV *Comma-Separated Values*

CNN *Convolutional Neural Network*

DL *Deep Learning*

DRNN *Disconnected Recurrent Neural Network*

DA *Data Augmentation*

ELMo *Embeddings from Language Models*

FP Falsos Positivos

FN Falsos Negativos

GPT *Generative Pre-Trained Transformer*

GV *Global Vectors*

GPU *Graphics Processing Unit*

HTML *HyperText Markup Language*

HTTP *Hypertext Transfer Protocol*

IoT *Internet of Things*

KNN *K-Nearest Neighbors*

LSTM *Long Short-Term Memory*

LLM *Large Language Models*

MMR *Maximal Marginal Relevance*

MC Matriz de Confusão

NB *Naive Bayes*

NLTK *Natural Language Toolkit*

PLN Processamento de Linguagem Natural

PDF *Portable Document Format*

POO Programação Orientada a Objetos

RNN *Recurrent Neural Network*

RL Regressão Logística

RN Rede Neural

RSR *Random Synonym Replacement*

RWS *Random Word Swapping*

SI Sistemas de Informação

SVM *Support Vector Machine*

TNT-LLM *Taxonomy and Text classification using Large Language Model*

TF-IDF *Term Frequency – Inverse Document Frequency*

TI Tecnologia da Informação

TPU *Tensor Processing Unit*

WEB *World Wide Web*

WE *Word Embedding*

WR *Word Representation*

VP Verdadeiros Positivos

VN Verdadeiros Negativos

Sumário

Capítulo 1

Introdução

A busca por informações na contemporaneidade demanda não apenas esforço, habilidades e tempo por parte do usuário, mas também se torna desafiadora devido aos incessantes avanços e mudanças tecnológicas. A acessibilidade a uma vasta quantidade de informações por meio de recursos tecnológicos, embora amplie as opções disponíveis, frequentemente resulta em sobrecarga informacional, podendo, por vezes, prejudicar a eficiência da busca e até mesmo propiciar desinformação. Em particular, órgãos públicos como por exemplo, defensorias que lidam com grandes volumes de textos jurídicos, textos jurídicos genéricos ou específicos, enfrentam dificuldades na classificação eficiente dessas informações, o que compromete a agilidade e a qualidade dos serviços prestados.

Com o contínuo progresso tecnológico, a simplificação do processo de pesquisa na internet e a obtenção eficiente de informações textuais tornam-se objetivos alcançáveis por meio da automação. As páginas *World Wide Web* (WEB), como fontes de dados amplamente utilizadas, deram origem à criação de diários oficiais por instituições públicas, configurando-se como ambientes para armazenamento de dados e informações (ZHANG, 2022). Neste contexto, técnicas avançadas de PLN e AP, como os modelos BERT, emergem como ferramentas promissoras para a classificação de textos jurídicos genérico ou especifico, oferecendo soluções mais precisas e eficientes em comparação com abordagens tradicionais.

Paralelamente à otimização de processos empresariais e industriais, a incorporação da Tecnologia da Informação (TI) nos órgãos públicos apresenta-se como um caminho para evolução. A utilização da TI possibilita a otimização de tarefas, reduzindo tanto o tempo quanto o esforço dos servidores públicos em suas atividades. Órgãos como as defensorias públicas, que demandam atendimento ao público, fazem uso de diversos SI para gerenciamento de informações e processos internos. Este trabalho concentra-se na aplicação de técnicas avançadas de PLN e AP, especificamente utilizando modelos BERT, para aprimorar a classificação de textos jurídicos específicos nestes órgãos, diferenciando-se de abordagens tradicionais ao explorar modelos de última geração.

A crescente demanda pelo uso de recursos tecnológicos nas repartições públicas im-

plica um investimento significativo de tempo e dedicação dos servidores na execução de tarefas (RANGEL et al., 2020). A automação de processos com fluxos bem definidos pode resultar em impactos positivos, como economia de tempo e esforço dos servidores, bem como economia de insumos, como papel, canetas e impressões (FERREIRA, 2019). A adoção de modelos avançados de PLN, como proposto neste trabalho, trará benefícios diretos, melhorando a eficiência operacional, reduzindo a carga de trabalho dos servidores e oferecendo suporte à tomada de decisões, além de potencialmente impactar positivamente a qualidade dos serviços públicos e a satisfação dos cidadãos.

Outro desafio enfrentado em alguns órgãos públicos é a necessidade de capacitar e treinar seus servidores para a transição dos processos físicos para o meio digital (FERREIRA, 2019). O avanço tecnológico cria resistência em parte dos servidores que persistem no trabalho manual, o que, por vezes, se revela improdutivo. Nesse contexto, a sobrecarga de informações e as rápidas mudanças tecnológicas tornam-se desafios para determinados usuários ao acessar e obter informações textuais essenciais para alimentar sistemas administrativos no âmbito do próprio órgão público. A solução proposta neste trabalho responde diretamente a automação de processos internos, oferecendo uma ferramenta que simplifica o processamento de informações e incentiva a adoção de tecnologias digitais.

O estudo de (RANGEL et al., 2020) destaca a demanda de tempo e esforço por parte dos usuários, ressaltando a necessidade de incorporação de recursos tecnológicos nas instituições públicas. Essa dedicação de tempo e esforço, frequentemente destinada ao acesso à portais *web*, obtenção de informações textuais, identificação e classificação de dados relevantes, é crucial para fornecer serviços, desenvolver atividades cotidianas, alimentar sistemas e gerar informações de importância interna ou externa ao órgão público (RANGEL et al., 2020). Diante disso, este trabalho visa contribuir para a otimização desses processos por meio da aplicação de técnicas avançadas de PLN e AP, reforçando a importância da pesquisa para automatizar processos internos, economizar tempo e embasar a tomada de decisão de forma mais eficiente, realização da classificação automática de informações relevantes contidas em portarias e atos administrativos, ou seja, documentos jurídicos específicos utilizados na rotina de um órgão público.

1.1 Motivação

Atualmente, estamos em uma era digital em que a tecnologia está cada vez mais integrada ao cotidiano das pessoas, independentemente de sua localização ou área profissional. A Indústria 4.0, caracterizada por inovações tecnológicas nos campos de automação, controle e tecnologia da informação, incluindo sistemas, Internet das Coisas, ou do inglês *Internet of Things* (IoT) e Internet dos Serviços, tem impulsionado mudanças significativas em diversos setores.

No contexto das instituições públicas, os processos de produção e prestação de servi-

ços tendem a se tornar cada vez mais eficientes, autônomos e customizáveis. A grande maioria dessas instituições utiliza documentos textuais para registrar e gerir informações, armazenando-as em portais WEB. Órgãos como as defensorias públicas fazem uso de diários oficiais para disponibilizar informações jurídicas relevantes, por exemplo, portarias e atos administrativos. Contudo, o volume crescente de informações textuais torna a gestão e a classificação manual desses documentos uma tarefa onerosa e suscetível a erros.

Pesquisas na área de classificação automática de textos têm buscado soluções para otimizar esse processo como evidenciado nos trabalhos de (MIAO et al., 2018) e (WU; LIU; WANG, 2020). No entanto, ainda existem desafios significativos a serem superados. Alguns classificadores não alcançam desempenho satisfatório, especialmente ao lidar com textos jurídicos complexos. Outros modelos enfrentam dificuldades devido à necessidade de um tratamento de dados eficiente para aprimorar a representação textual (MULAHUWAISH et al., 2020). Além disso, muitos dos modelos existentes não são otimizados para o contexto específico de documentos jurídicos em língua portuguesa.

Diante desses desafios, a motivação deste trabalho reside na necessidade de aprimorar a classificação automática de textos jurídicos específico em órgãos públicos, visando otimizar processos, reduzir a carga de trabalho dos servidores e melhorar a eficiência na gestão de informações. A aplicação de técnicas avançadas de PLN e AP, como os modelos BERT, apresenta-se como uma abordagem promissora para superar as limitações dos métodos atuais.

Assim, pretende-se contribuir para as pesquisas científicas em PLN e AP, com esforços concentrados em estratégias de tratamento de dados que aperfeiçoem a representação da base de dados utilizada para treinar classificadores automáticos de texto. Com isso, espera-se não apenas melhorar o desempenho dos modelos de classificação, mas também oferecer uma solução prática que possa ser implementada nas defensorias públicas e outros órgãos similares.

1.2 Objetivos e desafios da pesquisa

O objetivo principal deste trabalho foi investigar e propor a aplicação de um modelo baseado em aprendizado profundo, utilizando técnicas avançadas de processamento de linguagem natural como o BERT, para a classificação automática de textos jurídicos específicos em órgãos públicos, visando superar a abordagem e análise manual.

Para atingir o objetivo geral desta pesquisa, foram utilizados modelos de classificadores baseados em Rede Neural (RN) produndas como *Recurrent Neural Network* (RNN), *Long Short-Term Memory* (LSTM) e BERT. Dessa forma, este objetivo principal foi fragmentado em objetivos específicos listados abaixo.

1.2.1 Objetivos específicos

1. Coletar e preparar uma base de dados de textos jurídicos específicos provenientes do diário oficial de defensoria pública para permitir o treinamento eficaz dos modelos de aprendizado profundo.

2. Propor aplicação de um modelo de classificação baseado em AP, adaptado para o contexto de textos jurídicos específicos em língua portuguesa, e avaliar sua eficácia em relação a modelos existentes na literatura.

3. Analisar o impacto de técnicas de pré-processamento e tratamento de dados no desempenho dos modelos de classificação de textos jurídicos específicos.

4. Propor a implementação prática da automação de processos internos em órgãos públicos, visando à otimização de processos e suporte à tomada de decisões.

1.3 Hipótese

Diante dos desafios identificados pela crescente utilização de páginas *web* para armazenar dados, grande volume de dados, pela necessidade de filtragem manual de informações textuais nessas páginas e pelas dificuldades na classificação manual de textos jurídicos específicos em órgãos públicos, este trabalho propôs testar a seguinte hipótese:

1. Utilização de modelos de aprendizado profundo e a aplicação de técnicas de processamento de linguagem natural no pré-processamento dos dados melhora significativamente o desempenho dos classificadores de textos jurídicos específicos no idioma português.

1.3.1 Justificativa das hipóteses

Trabalhos anteriores utilizaram outras abordagens e a abordagem deste trabalho difere ou avança em relação à eles:

1. Trabalhos como o de (PIVETTA; MERGEN; KEPLER, 2013) focaram na classificação de textos sem um pré-processamento nos dados. Espera-se que, ao aplicar técnicas de PLN no pré-processamento, os resultados sejam significativamente melhorados.

2. A automatização completa do processo, desde a coleta até a classificação, pode reduzir o tempo e o esforço necessários, abordando as limitações dos métodos manuais destacados em (RANGEL et al., 2020).

3. Enquanto estudos como os de (MIAO et al., 2018) e (MULAHUWAISH et al., 2020) concentraram-se na etapa de classificação, este trabalho enfatiza a importância de uma base de dados bem estruturada e limpa para otimizar o desempenho dos modelos.

1.4 Contribuições

As contribuições desta pesquisa foram:

1. O desenvolvimento de um modelo baseado no BERT, ajustado para lidar com a complexidade e especificidade da linguagem jurídica específica em língua portuguesa. Este modelo superou a abordagem tradicional baseada em *Term Frequency – Inverse Document Frequency* (TF-IDF) e obteve F1-*Score* melhor que a RNN e LSTM, alcançando uma acurácia de 99,27% na classificação de textos jurídicos específicos extraídos de portarias e atos administrativos.

2. Realização de uma avaliação do desempenho das arquiteturas RNN, LSTM e BERT na tarefa de classificação de textos jurídicos específicos. Os resultados desta análise fornecem informações dos termos mais trabalhados e utilizados nesses documentos jurídicos administrativos nesse contexto de dados.

3. Desenvolvimento de uma abordagem metodológica de automação do processo desde a coleta e pré-processamento dos dados até a classificação das informações. Esta metodologia pode ser replicada por outras instituições públicas que desejem automatizar processos similares, contribuindo para a padronização e eficiência na adoção dessas tecnologias.

4. Implementação um processo piloto que automatiza a classificação de textos de portarias e atos administrativos no contexto jurídico, demonstrando na prática como a aplicação de técnicas de PLN e AP pode reduzir a carga de trabalho dos servidores, otimizar processos internos e melhorar a qualidade dos serviços prestados à sociedade. Este sistema evidencia o potencial impacto positivo da tecnologia na modernização da administração pública.

1.5 Organização da dissertação

Esta dissertação está organizada em seis capítulos que abordam sequencialmente os aspectos fundamentais da pesquisa. No Capítulo 1, apresenta-se a introdução ao assunto abordado, contextualizando o tema, expondo a motivação, os objetivos, as hipóteses, as contribuições e delineando a estrutura geral do trabalho. O Capítulo 2 traz a fundamentação teórica, onde são discutidos os conceitos essenciais de processamento de linguagem

natural, aprendizado de máquina, aprendizado profundo e suas aplicações em classificação de textos. No Capítulo 3, são apresentados os trabalhos relacionados, explorando as abordagens tradicionais e estudos correlatos que servem como base para a pesquisa. O Capítulo 4 detalha a metodologia empregada, incluindo os métodos e técnicas utilizados, a coleta de dados, o pré-processamento, o desenvolvimento dos modelos, as ferramentas empregadas e as etapas que serão seguidas para atingir os objetivos propostos. O Capítulo 5 é dedicado aos experimentos e à análise dos resultados, apresentando os experimentos realizados, a análise dos resultados obtidos e a comparação entre os modelos propostos. Finalmente, o Capítulo 6 encerra o trabalho com a conclusão, resumindo as principais contribuições da pesquisa, destacando as limitações do estudo e sugerindo direções para trabalhos futuros.

Capítulo 2

Fundamentação Teórica

2.1 Classificação automática de textos

A classificação automática de textos, que envolve associar textos em linguagem natural a rótulos pré-definidos, é uma área crucial que combina extração de informação e aprendizado de máquina. Segundo (FIGUEIREDO, 2008), essa técnica tem aplicações variadas, incluindo indexação automática de textos, identificação de autores e filtragem de e-mails. Recentemente, (ZHANG, 2022) expandiu essas aplicações, demonstrando que algoritmos de aprendizado profundo, como Transformers, podem melhorar significativamente a precisão na classificação de grandes volumes de dados textuais.

O aumento dos recursos e o uso generalizado da internet para a disponibilização de informações resultaram em um crescente volume de dados a serem armazenados e acessados por meio de páginas web. Esses dados são frequentemente organizados em documentos textuais, nos formatos *HyperText Markup Language* (HTML) e PDF, sendo alvos principais de robôs de busca e ferramentas de recuperação de informação. Essas ferramentas desempenham funções como busca por informações textuais e filtragem de textos com base em conteúdos específicos. Uma técnica associada à recuperação de documentos é a classificação automática de textos, que tem como objetivo identificar a qual categoria pré-definida um documento de texto pertence, agrupando documentos semanticamente relacionados (SEBASTIANI, 2002).

2.2 Aprendizado de máquina

O aprendizado de máquina é uma área de pesquisa que visa capacitar computadores a reconhecer padrões e executar tarefas com base na utilização de dados, permitindo a tomada automática de decisões. Entre as diferentes tarefas realizáveis, a classificação é a mais utilizada e conhecida. Seu objetivo é simples: treinar um modelo computacional para que seja capaz de atribuir automaticamente classes (ou "rótulos de classe") a objetos cujas classes sejam desconhecidas (GONçALVES, 2018). No aprendizado de máquina,

podemos considerar que a abordagem clássica do AM compreende três tipos principais de aprendizado: supervisionado, não supervisionado e por reforço, conforme (RANGEL et al., 2020).

Além da classificação, o aprendizado de máquina abrange uma variedade de tarefas, como regressão, agrupamento (*clustering*), detecção de anomalias e redução de dimensionalidade. A regressão é utilizada para prever valores contínuos, enquanto o agrupamento busca identificar estruturas ou padrões ocultos em dados não rotulados. No aprendizado supervisionado, modelos são treinados com dados rotulados para fazer previsões ou classificações sobre novos dados. Já no aprendizado não supervisionado, o modelo tenta inferir a estrutura subjacente dos dados sem referências externas. O aprendizado por reforço envolve a interação de um agente com um ambiente, aprendendo a tomar decisões ótimas através de recompensas e penalidades (SUTTON; BARTO, 2018). Essas técnicas têm sido amplamente aplicadas em campos como medicina, finanças e robótica, contribuindo para avanços significativos na resolução de problemas complexos (JORDAN; MITCHELL, 2015).

2.3 Aprendizado profundo

O aprendizado profundo, é um subcampo do aprendizado de máquina que se concentra em algoritmos inspirados na estrutura e função do cérebro, chamados de redes neurais artificiais. O termo (profundo) refere-se ao uso de múltiplas camadas de redes neurais para aprender representações de dados com diferentes níveis de abstração. O aprendizado profundo tem a capacidade de modelar padrões complexos em grandes volumes de dados, tornando-o especialmente eficaz em áreas como visão computacional, processamento de linguagem natural, reconhecimento de fala, entre outras (GOODFELLOW; BENGIO; COURVILLE, 2016).

Além das arquiteturas básicas, o aprendizado profundo engloba diversas técnicas e modelos avançados que permitem o tratamento de dados não estruturados e a resolução de problemas complexos. Por exemplo, as *Convolutional Neural Network* (CNN) são amplamente utilizadas em tarefas de visão computacional devido à sua capacidade de extrair características espaciais de imagens. Já as redes RNNs e suas variantes, como as redes LSTMs e as *Bidirectional Recurrent Neural Networks* (BRNN), são eficazes no processamento de sequências temporais, sendo aplicadas em reconhecimento de fala e tradução automática. Além disso, técnicas como o aprendizado por reforço profundo combinam redes neurais com algoritmos de aprendizado por reforço, permitindo que agentes aprendam a tomar decisões sequenciais em ambientes complexos (PLAAT, 2022). Esses avanços têm expandido significativamente o alcance do aprendizado profundo, impulsionando inovações em diversas áreas da ciência e tecnologia.

2.4 Overfitting e underfitting

Ajuste excessivo, ou do inglês, *overfitting* é um fenômeno em aprendizado de máquina em que o modelo se ajusta tão bem aos dados de treinamento que acaba capturando não apenas os padrões reais, mas também o ruído e as particularidades específicas desses dados. Isso resulta em alta acurácia no treinamento, mas desempenho insatisfatório ao ser aplicado em novos dados, como os conjuntos de validação ou teste. Modelos que sofrem de *overfitting* são geralmente complexos, contendo muitos parâmetros, camadas ou neurônios em comparação ao volume de dados disponível. Para mitigar o problema, estratégias como regularização (L1, L2, *dropout*), interrupção precoce do treinamento (*early stopping*) e redução da complexidade do modelo são amplamente utilizadas. A coleta de mais dados ou o uso de aumento de dados (*data augmentation*) também pode ajudar na generalização do modelo (CHOLLET, 2021).

Ajuste insuficiente, ou do inglês, *underfitting* ocorre quando o modelo é incapaz de aprender adequadamente os padrões subjacentes dos dados de treinamento, apresentando baixo desempenho tanto no treinamento quanto na validação. Esse problema surge geralmente quando o modelo é muito simples ou não teve tempo suficiente de treinamento para capturar a complexidade dos dados. Modelos subajustados falham em generalizar e não conseguem capturar nem mesmo as relações básicas entre as variáveis. A solução para o *underfitting* envolve aumentar a complexidade do modelo, adicionando mais camadas ou neurônios, treinar por mais épocas e, muitas vezes, aplicar engenharia de atributos para garantir que os dados sejam representados de forma mais adequada para o aprendizado (CHOLLET, 2021).

2.5 Processamento de linguagem natural

O processamento de linguagem natural, investiga o uso de máquinas computacionais para processar ou compreender a língua humana, com o propósito de realizar atividades úteis. Envolve a construção de mecanismos artificiais (computadores) capazes de entender a linguagem natural para realizar tarefas ou aplicações próximas ao entendimento humano. O PLN é considerado um campo interdisciplinar que combina linguística computacional, computação científica, ciência cognitiva e inteligência artificial. De forma geral, o PLN é um método computacional de análise de textos realizado por computadores, englobando um conjunto de teorias e tecnologias (DENG; LIU, 2018).

Conforme mencionado por(DENG; LIU, 2018), o PLN pode ser aplicado em diversas áreas, como reconhecimento de fala, compreensão da linguagem falada, sistemas de diálogo, análise lexical, tradução automatizada de áudio e texto, sistemas de recuperação de informações, sistemas de perguntas e respostas, análise de sentimentos, geração de linguagem natural e idioma natural. Apesar dos avanços nas aplicações citadas, os

modelos de aprendizado de máquina e aprendizado profundo para PLN ainda apresentam desafios. Muitas vezes, esses modelos não possuíam capacidade suficiente para absorver grandes quantidades de dados de treinamento, tanto devido a limitações nos algoritmos quanto nas infraestruturas computacionais. No entanto, ao longo do tempo, isso mudou, dando origem à terceira onda de PLN, impulsionada pelo novo paradigma de aprendizado profundo conforme a literatura pertinente.

Nos últimos anos, o uso de arquiteturas híbridas, como a combinação de RNNs e CNNs, tem sido uma tendência emergente para melhorar o desempenho em tarefas de classificação de texto. Essas abordagens permitem a captura eficiente de relações locais e globais em documentos textuais, aprimorando a compreensão semântica e a precisão dos modelos. A aplicação dessas técnicas em áreas como saúde pública, para analisar grandes volumes de textos, exemplifica o poder do PLN em fornecer *insights* valiosos para tomadas de decisão em tempo real (LAVANYA; SASIKALA, 2021). Esse desenvolvimento representa uma mudança significativa na forma como os dados textuais são processados e utilizados em diversas indústrias.

Além disso, o PLN experimentou um avanço significativo com a adoção de técnicas de aprendizado profundo, como as RNNs e as redes LSTMs. Essas técnicas têm sido aplicadas com sucesso em uma ampla gama de aplicações de PLN, como tradução automática, análise de sentimentos e sistemas de perguntas e respostas. A combinação de modelos de DL com técnicas de pré-processamento robustas, como a remoção de ruídos e tokenização eficiente, tem proporcionado resultados mais precisos e relevantes, principalmente em cenários de grande volume de dados não estruturados, como os provenientes de redes sociais e sistemas de saúde (SHARMA; DEVI, 2023). Esse progresso reflete o potencial crescente do PLN para lidar com dados textuais complexos em escala.

2.6 Representação vetorial de textos

Word Embedding (WE) é uma técnica no processamento de linguagem natural que transforma palavras em vetores numéricos, permitindo que máquinas compreendam e processem texto de maneira eficaz. Ao representar palavras como vetores em um espaço de alta dimensão, essa abordagem captura relações semânticas e sintáticas entre as palavras, posicionando termos semelhantes próximos uns dos outros nesse espaço vetorial. Para realizar essa transformação, utilizam-se técnicas como *Bag of Words* (BoW) e TF-IDF, que contabilizam a frequência das palavras, e métodos mais avançados como *Word2Vec*, *GloVe*, *fastText* e *ELMo*, que consideram o contexto e o significado profundo das palavras em um corpus de texto.

Esses *embeddings* são aplicados em diversas áreas, como tradução automática, análise de sentimentos e sistemas de recomendação, trazendo vantagens como a redução da dimensionalidade dos dados e a melhoria na captura de similaridades semânticas. No

entanto, também apresentam desafios, como a necessidade de grandes volumes de dados para treinamento e a dificuldade em representar palavras raras ou com múltiplos significados. O avanço dessas técnicas tem sido fundamental para o desenvolvimento de aplicações mais inteligentes e contextualmente conscientes em PLN (CHOLLET, 2021).

2.6.1 Bag of words

BoW é uma técnica básica de representação textual que transforma textos em vetores de frequência de palavras, ignorando a ordem e a estrutura sintática. Cada documento é representado pela contagem de ocorrências de cada palavra presente no vocabulário. É adequado para conjuntos de dados menores e tarefas simples de classificação, onde a simplicidade e a interpretabilidade são desejadas. BoW é eficaz em domínios específicos, pois pode capturar termos relevantes para um determinado campo. No entanto, não considera a complexidade da linguagem, como contexto ou polissemia, o que limita sua capacidade de lidar com textos mais sofisticados (JURAFSKY; MARTIN, 2024).

2.6.2 TF-IDF

TF-IDF é uma técnica que pondera a frequência de uma palavra em um documento com a raridade dessa palavra em todo o corpus, destacando termos importantes para um documento específico. Recomenda-se seu uso em conjuntos de dados de tamanho médio, onde se busca equilibrar entre simplicidade e a capacidade de capturar termos relevantes. TF-IDF é eficaz em domínios específicos, pois enfatiza termos significativos dentro de um campo particular. No entanto, como não captura relações semânticas ou contexto, sua eficácia é limitada em lidar com a complexidade da linguagem, especialmente em textos ambíguos ou polissemânticos (JURAFSKY; MARTIN, 2024).

2.6.3 Word2Vec

Word2Vec é uma técnica de aprendizado não supervisionado que gera representações vetoriais densas para palavras, capturando relações semânticas e sintáticas em um espaço contínuo. Ela utiliza modelos de redes neurais rasas, como o *Continuous Bag-of-Words (CBOW)* e o *Skip-Gram*, para prever palavras com base em seu contexto. Recomenda-se seu uso em conjuntos de dados grandes, pois a qualidade das *embeddings* melhora com mais dados de treinamento disponíveis. *Word2Vec* é adequado tanto para domínios gerais quanto específicos, mas pode requerer ajustes finos para terminologias especializadas. Em termos de complexidade da linguagem, funciona bem com idiomas de morfologia menos complexa, podendo enfrentar desafios em línguas altamente flexionais (JURAFSKY; MARTIN, 2024).

2.6.4 GloVe

GloVe é uma técnica que combina abordagens baseadas em contagem com métodos preditivos para aprender representações vetoriais de palavras. Ela utiliza estatísticas globais de coocorrência para capturar relações semânticas entre palavras, resultando em *embeddings* que refletem similaridades semânticas e sintáticas. *GloVe* é recomendado para conjuntos de dados grandes devido à necessidade de estatísticas robustas de coocorrência. É eficaz em domínios gerais e pode ser adaptado para domínios específicos com dados suficientes. Quanto à complexidade da linguagem, *GloVe* é adequado para idiomas com morfologia menos complexa e pode ter limitações em capturar nuances contextuais (JURAFSKY; MARTIN, 2024).

2.6.5 FastText

FastText é uma técnica de *embeddings* de palavras que estende o *Word2Vec* ao incorporar informações de subpalavras, permitindo capturar relações morfológicas e lidar com palavras raras ou não vistas. Utiliza modelos de redes neurais para aprender representações não apenas de palavras inteiras, mas também de n-gramas de caracteres. É recomendado para conjuntos de dados grandes, mas é mais eficiente que modelos similares devido à sua arquitetura otimizada. *FastText* é adequado tanto para domínios gerais quanto específicos, especialmente em idiomas com morfologia rica. Em termos de complexidade da linguagem, lida bem com variações morfológicas e é eficaz em capturar nuances em línguas altamente flexionais (WEHRMANN; KOLLING; BARROS, 2019).

2.6.6 ELMo

Embeddings from Language Models (ELMo) é uma técnica de *embeddings* contextuais que gera representações dinâmicas de palavras considerando todo o contexto em que elas aparecem. Utiliza redes neurais profundas baseadas em modelos de linguagem bidirecionais para capturar nuances semânticas e sintáticas. É recomendado para conjuntos de dados grandes devido à complexidade computacional envolvida no treinamento dos modelos. ELMo é eficaz em domínios gerais e pode ser adaptado para domínios específicos com dados suficientes. Em relação à complexidade da linguagem, é particularmente eficaz em lidar com idiomas de morfologia complexa e ambiguidades contextuais, capturando significados dependentes do contexto (JURAFSKY; MARTIN, 2024).

2.7 Aplicações envolvendo PLN

Atualmente, o processamento de linguagem natural está presente em diversas aplicações cotidianas. Na maioria das vezes, as aplicações que utilizam processamento de textos

são fortes candidatas a empregar métodos de PLN. Essas aplicações conseguem extrair informações de centenas de textos armazenados em uma base de dados. Um exemplo cotidiano que utiliza técnicas de PLN são os *chatbots*, o *Chat Generative Pre-Trained Transformer* (GPT) entre outras aplicações, como recuperação da informação, extração de informação, sumarização automática, tradução de texto, sistemas de diálogo, reconhecimento de fala e compreensão da linguagem natural.

Com o aumento da demanda por sistemas de informações, diversas aplicações necessitam realizar tarefas que envolvem processamento de texto. Por exemplo, a classificação automática de documentos, extração de conhecimentos em dados brutos em diversas áreas, o reconhecimento de caracteres escritos manualmente, entre outras atividades. No aprendizado supervisionado, o modelo é treinado a partir de dados rotulados, com o objetivo de prever rótulos para novos dados. Já o aprendizado não supervisionado busca explorar recursos dos dados sem rótulos, enquanto o aprendizado por reforço baseia-se em tentativa e erro, aprendendo por meio da interação com o ambiente.

O PLN tem sido utilizado em soluções de automação residencial, onde os usuários podem controlar dispositivos por meio de comandos de voz. A integração do PLN com a Internet das Coisas permitiu o desenvolvimento de casas inteligentes, nas quais aparelhos podem ser gerenciados por chatbots que entendem e respondem a comandos naturais. Esses sistemas têm mostrado benefícios significativos para pessoas com deficiência, proporcionando maior independência e conforto (MOHANA et al., 2022). Também há aplicações relevantes no campo da educação em saúde, onde o PLN auxilia na disseminação de informações e no monitoramento de surtos de doenças infecciosas (JIANG et al., 2021). Além disso, aplicações que utilizam o PLN têm sido amplamente aplicadas em diferentes áreas da saúde, com destaque para o uso em sistemas de suporte à decisão clínica e extração de informações de registros eletrônicos de saúde. A capacidade do PLN de lidar com grandes volumes de dados não estruturados, como anotações médicas, têm permitido avanços na identificação de diagnósticos e tratamentos mais precisos, além de facilitar a análise de dados clínicos em tempo real (SETT; SINGH, 2024). Esses sistemas são particularmente úteis na detecção precoce de doenças, como a COVID-19, e na melhoria da eficiência em ambientes hospitalares.

2.8 Arquiteturas de redes recorrentes

Nos últimos cinco anos, a literatura em ciência da computação tem destacado o papel fundamental das RNNs no processamento de dados sequenciais, como séries temporais, reconhecimento de fala e PLN. As RNNs tradicionais, no entanto, sofrem de limitações ao tentar capturar dependências de longo prazo em sequências devido ao problema do gradiente desaparecido, o que compromete seu desempenho em tarefas complexas. Para mitigar essa limitação, foram desenvolvidas variações arquiteturais mais robustas, como as

redes LSTMs, que introduzem mecanismos de portas para controlar o fluxo de informações relevantes e irrelevantes ao longo da sequência, melhorando a retenção de dependências distantes e garantindo maior eficiência no treinamento de modelos complexos (SOUZA; SILVA, 2021).

As redes LSTMs, devido à sua capacidade de lidar com o problema de dependências de longo prazo, têm sido amplamente utilizadas em aplicações que exigem a compreensão contextual de grandes sequências de dados, como tradução automática, *chatbots* e previsão de séries temporais. A adição de portas de entrada, saída e esquecimento permite que as LSTMs filtrem e armazenem informações importantes ao longo do tempo, superando as limitações das RNNs simples. Estudos recentes demonstram que as LSTMs continuam a ser a escolha preferida em muitas tarefas de PLN e visão computacional, sendo aprimoradas por abordagens híbridas com outras técnicas, como *Attention Mechanism* e *Transformer Models*, que visam combinar o melhor dos dois mundos (OLIVEIRA et al., 2022).

2.9 Processamento de linguagem natural com BERT

O bidirectional encoder representations from transformers é um modelo de linguagem desenvolvido para pré-treinamento de representações bidirecionais profundas a partir de dados não rotulados, permitindo a captação de relações contextuais em ambos os sentidos (da esquerda para a direita e da direita para a esquerda) em uma sentença. Esse modelo que utiliza o *Transformer*, um mecanismo de atenção que captura as relações contextuais entre palavras ou subpalavras em um determinado texto, foi projetado para melhorar a compreensão do texto em uma ampla gama de tarefas de processamento de linguagem natural, utilizando camadas transformadoras para capturar a complexidade do contexto das palavras de maneira mais eficaz (DEVLIN et al., 2019).

O BERT tem se mostrado extremamente eficaz em várias tarefas de processamento de linguagem natural, como análise de sentimento, onde captura nuances de significado em frases; resposta a perguntas, identificando trechos relevantes que respondem a uma questão; reconhecimento de entidades nomeadas, classificando elementos específicos do texto como pessoas e organizações; tradução automática, aprimorando o contexto da tradução; sumarização de textos, extraindo informações essenciais; e classificação de textos, categorizando conteúdo em diferentes tópicos com alta precisão (DEVLIN et al., 2019).

2.10 Tecnologias e ferramentas

Nesta seção, serão apresentadas as principais tecnologias e ferramentas utilizadas no desenvolvimento desta pesquisa. Elas são: *Python*, Pandas, *SpaCy*, *PyTorch*, *Scikit-learn* e *Google Colaboratory*. Essas ferramentas desempenham papéis fundamentais em

diversas etapas do processo, desde a manipulação e análise de dados, processamento de linguagem natural até a implementação de modelos de aprendizado profundo. A escolha dessas tecnologias se justifica por sua robustez, facilidade de uso e ampla adoção tanto em ambientes acadêmicos quanto em aplicações industriais, proporcionando flexibilidade e eficiência no desenvolvimento de soluções inovadoras em ciência de dados e aprendizado profundo de acordo com a literatura pertinente.

2.10.1 Python

Python é uma linguagem de programação de alto nível, interpretada e conhecida por sua simplicidade e versatilidade. Por oferecer uma sintaxe clara e intuitiva, é amplamente utilizada tanto por iniciantes quanto por profissionais experientes em diversas áreas, como ciência de dados, automação, e desenvolvimento *web*. Além disso, Python conta com uma vasta comunidade e uma rica coleção de bibliotecas que facilitam tarefas complexas, como análise de dados (Pandas) e aprendizado de máquina (*Scikit-learn*) (MCKINNEY, 2022).

2.10.2 Pandas

Pandas é uma biblioteca fundamental para manipulação e análise de dados em *Python*, oferecendo estruturas como *DataFrames* que facilitam a organização e processamento de grandes conjuntos de dados. Sua principal vantagem está na facilidade de manipulação de dados tabulares e integração com outras bibliotecas como *NumPy* e *Matplotlib*. No entanto, Pandas pode consumir muita memória, especialmente ao lidar com grandes volumes de dados (MCKINNEY, 2022).

2.10.3 Spacy

SpaCy é uma biblioteca *open-source* de PLN desenvolvida para lidar com grandes volumes de texto de maneira eficiente e rápida. Voltada para aplicações em produção, *SpaCy* se destaca por fornecer modelos pré-treinados e funcionalidades prontas para tarefas como análise sintática, reconhecimento de entidades nomeadas, lematização e vetorização de palavras. A biblioteca foi projetada para ser prática e acessível, com uma *Application Programming Interface* (API) intuitiva e suporte a integração com outras ferramentas do ecossistema *Python*, como *TensorFlow* e *PyTorch* (VASILIEV, 2020).

Uma das principais vantagens do *SpaCy* é seu desempenho otimizado por meio de *Cython*, o que permite processar texto a uma velocidade consideravelmente superior a outras bibliotecas tradicionais como *Natural Language Toolkit* (NLTK). A biblioteca é amplamente utilizada em aplicações de assistentes virtuais, *chatbots*, análise de sentimentos e sistemas de busca. Além disso, *SpaCy* oferece suporte robusto para treinamento de modelos personalizados, facilitando a adaptação para casos específicos, como aplicações biomédicas ou jurídicas (VASILIEV, 2020).

2.10.4 Pytorch

PyTorch é uma estrutura para aprendizado profundo que permite a criação de modelos complexos de forma flexível e dinâmica. Utilizada amplamente em pesquisa e produção, *PyTorch* se destaca pelo suporte a computação em *Graphics Processing Unit* (GPU) e pelo uso intuitivo de gráficos dinâmicos. Porém, seu aprendizado pode ser desafiador para iniciantes e a documentação, apesar de completa, pode não ser suficiente para casos muito específicos (STEVENS; ANTIGA; VIEHMANN, 2020).

2.10.5 Scikit-learn

Scikit-learn é uma biblioteca abrangente para aprendizado de máquina em *Python*, oferecendo implementações eficientes de algoritmos comuns, como regressão linear e árvores de decisão. Ela é ideal para aplicações tradicionais de aprendizado supervisionado e não supervisionado. No entanto, pode não ser adequada para tarefas mais complexas que envolvem redes neurais profundas, sendo recomendável utilizar frameworks como *PyTorch* nesses casos (MCKINNEY, 2022).

2.10.6 Google colaboratory

O *Google Colaboratory*, conhecido como Colab, é uma plataforma gratuita baseada na nuvem que permite a criação e execução de código *Python* diretamente no navegador, sem necessidade de instalação ou configuração. Originalmente desenvolvido pelo *Google Research*, o Colab é especialmente útil para aprendizado de máquina, análise de dados e atividades educacionais. Além de suportar *Python*, ele pode ser adaptado para rodar outras linguagens, como R, Julia e *Swift*, proporcionando versatilidade para diferentes tipos de projetos (BISONG, 2019).

Com o Colab, os usuários têm acesso gratuito a aceleradores de GPU e *Tensor Processing Unit* (TPU), fundamentais para o treinamento de modelos que demandam alto poder computacional. A plataforma oferece ainda a integração com notebooks *Jupyter*, bibliotecas pré-instaladas, como *TensorFlow* e *Matplotlib*, e um ambiente colaborativo que permite o compartilhamento e edição simultânea de projetos de pesquisa e desenvolvimento, de forma semelhante ao *Google Docs*. Os *notebooks* são armazenados automaticamente no *Google Drive*, facilitando o acesso e a continuidade do trabalho (BISONG, 2019).

Capítulo 3

Trabalhos Relacionados

A classificação de textos é um campo que tem evoluído significativamente nas últimas décadas, impulsionado pelo crescimento exponencial de dados textuais e pela necessidade de processá-los de forma eficiente. Nesta seção, foram revisados trabalhos relacionados, organizados em três categorias: Técnicas Tradicionais de Classificação de Texto, Redes Neurais e Redes Neurais Profundas. Em cada categoria, discutimos o problema abordado, a metodologia empregada, os resultados obtidos e as contribuições e limitações de cada estudo. Ao final, conectamos esses trabalhos com a presente pesquisa, destacando sua relevância.

3.1 Técnicas tradicionais de classificação de texto

3.1.1 Aplicação de algoritmos clássicos

No contexto inicial da classificação de textos, algoritmos tradicionais de aprendizado de máquina desempenharam um papel fundamental. Em 2009, (ZHANG; NIU; NIE, 2009) abordaram o problema de aprimorar o algoritmo *K-Nearest Neighbors* (KNN) tradicional para a classificação de documentos coletados na *web*. O método tradicional atribui classes com base nos k vizinhos mais próximos, o que pode ser insuficiente diante da ambiguidade inerente à linguagem natural. Para enfrentar esse desafio, os autores propuseram o método fuzzy KNN, que incorpora a teoria dos conjuntos fuzzy para atribuir graus de pertinência a cada classe, refletindo a incerteza na classificação.

A metodologia envolveu o uso do TF-IDF para a seleção de características relevantes nos documentos, permitindo uma representação numérica eficaz. Os resultados mostraram que o fuzzy KNN superou tanto o KNN tradicional quanto o SVM, alcançando um F1-*score* de 76,38%, em comparação com 65,13% e 67,27%, respectivamente, dos métodos concorrentes. A principal contribuição desse estudo foi demonstrar que a inclusão de incerteza na classificação pode melhorar significativamente a precisão. Entretanto, uma limitação notável foi a redução na velocidade de classificação, tornando o método menos

adequado para grandes volumes de dados.

Em 2013, (PIVETTA; MERGEN; KEPLER, 2013) enfrentou o problema de classificar documentos militares do Exército Brasileiro, que exigem alto nível de precisão devido à sua natureza sensível. Utilizando o algoritmo *Naive Bayes* (NB), a metodologia incluiu uma base de treinamento composta por 200 boletins internos de 64 militares. Técnicas de pré-processamento, como seleção por janela deslizante e n-gramas, foram empregadas para extrair características significativas.

Os resultados indicaram que a combinação do NB com janela deslizante e n-gramas atingiu um F1-*score* de 76,7%, superando a abordagem baseada apenas em trigramas, que obteve 69,7%. A contribuição deste trabalho reside na eficácia do NB quando combinado com técnicas adequadas de extração de características. No entanto, a limitação principal é a dependência do desempenho em relação ao conjunto de dados específico, o que pode dificultar a generalização para outros domínios.

3.1.2 Uso de n-gramas e aprendizado de máquina

Com o aumento da disseminação de informações falsas na internet, a detecção de *fake news* tornou-se um problema crítico. Em 2017, (AHMED; TRAORE; SAAD, 2017) exploraram esse desafio, aplicando n-gramas e algoritmos de aprendizado de máquina para identificar conteúdo enganoso. A metodologia envolveu a extração de características utilizando n-gramas e a aplicação de algoritmos como o NB e a Regressão Logística (RL).

O estudo utilizou um conjunto de dados abrangente, contendo 25.200 notícias reais e falsas, permitindo uma análise robusta. Os resultados foram promissores, com uma precisão de 92%, demonstrando a robustez da abordagem para identificar *fake news* em diversos formatos e fontes. A principal contribuição foi evidenciar que técnicas tradicionais, quando combinadas com estratégias eficazes de extração de características, podem ser altamente eficientes na detecção de desinformação. Contudo, uma limitação é a possibilidade de que os modelos não capturem estratégias de desinformação emergentes, exigindo atualizações contínuas.

3.2 Redes neurais

3.2.1 Modelos recorrentes e convolucionais

A evolução do aprendizado profundo trouxe avanços significativos para a classificação de textos. Em 2018, (WANG, 2018) abordou o problema das limitações das RNNs tradicionais em capturar simultaneamente padrões locais e dependências de longo prazo em textos. Para enfrentar esse desafio, desenvolveu o modelo *Disconnected Recurrent Neural Network* (DRNN), que limita a transmissão de informações entre estados ocultos adja-

centes, permitindo a captura eficaz de padrões locais sem sacrificar as dependências de longo alcance.

A metodologia foi testada em sete bases de dados, incluindo 20 *Newsgroups* e *Reuters*, e os resultados mostraram um F1-*score* acima de 90%. A contribuição significativa deste trabalho foi demonstrar que o DRNN é particularmente útil para textos com alta variabilidade temática, onde a posição das palavras é crítica. No entanto, uma limitação é a complexidade computacional do modelo, que pode exigir recursos significativos para treinamento.

No mesmo ano, (WANG; KIM, 2018) enfrentaram o problema de lidar com textos de tamanhos variados e desordenados. Eles combinaram CNNs e LSTM em sua metodologia, aproveitando a capacidade das CNNs de extrair características locais e das LSTMs de capturar dependências sequenciais.

Testado em quatro bases de dados de referência, o modelo superou os métodos tradicionais em tarefas de análise de sentimentos e classificação temática. A contribuição foi demonstrar que a combinação de CNNs e LSTMs é uma solução promissora para problemas complexos de PLN. A limitação principal é a necessidade de um volume substancial de dados para treinar efetivamente o modelo, além do aumento do tempo de processamento.

3.2.2 Comparação dos algoritmos de classificação

Ainda em 2018, (MIAO et al., 2018) investigou qual algoritmo tradicional seria mais eficaz para a classificação de textos em uma base de notícias da Universidade Fudan. Abordando o problema de identificar o classificador mais adequado, a metodologia comparou os algoritmos KNN, NB e *Support Vector Machine* (SVM).

Os resultados mostraram que o SVM foi o mais eficaz, atingindo 95% de precisão e F1-*score*, destacando-se dos demais. A contribuição desse estudo foi reforçar a eficácia do SVM em tarefas de classificação textual e a importância de escolher o classificador apropriado para o tipo de dados. Contudo, a limitação reside na possível variabilidade do desempenho dependendo das características específicas do conjunto de dados.

Em 2020, (RANGEL et al., 2020) enfrentaram o problema de categorizar grandes volumes de dados textuais de portais governamentais. Utilizando os algoritmos Multinomial NB e SVM, a metodologia destacou a vetorização *bag of words* para representar os textos.

Os resultados indicaram que o SVM apresentou melhor desempenho, apesar de alguns termos importantes terem sido perdidos no pré-processamento. A contribuição foi evidenciar que, mesmo com técnicas simples de vetorização, o SVM pode ser eficaz na categorização de dados abertos. A limitação é a perda potencial de informações semânticas devido ao pré-processamento simplificado.

(WU; LIU; WANG, 2020), em 2020, abordou os desafios de processar textos com regionalismos e grafias variáveis. A metodologia combinou CNNs e RNNs para aproveitar as

vantagens de ambos os modelos. Os resultados demonstraram que redes neurais profundas são soluções mais eficazes para esses problemas complexos de PLN.

A contribuição foi destacar a adaptabilidade dos modelos profundos a diferentes contextos linguísticos. No entanto, a limitação está na necessidade de grandes volumes de dados e na complexidade computacional associada.

Em 2021, (CHERIF; MADANI; KISSI, 2021) propuseram uma técnica inovadora para classificação textual, enfrentando o problema de reduzir a complexidade computacional sem comprometer a precisão. A metodologia baseou-se em limiares e redução de dimensionalidade, eliminando a necessidade de técnicas de pré-processamento como *stemming*.

Os resultados mostraram um F1-*score* superior a 95% em bases como *BBC News* e *Reuters*. A contribuição foi demonstrar que é possível alcançar alta precisão com menor complexidade computacional. A limitação é que a eficácia pode diminuir em textos mais longos ou com maior variabilidade temática.

3.3 Redes neurais profundas

3.3.1 Aprendizado multitarefa e sumarização extrativa

Em 2022, (AGARWAL; XU; GRABMAIR, 2022) enfrentaram o problema de resumir textos jurídicos complexos com precisão comparável à de especialistas humanos. A metodologia combinou aprendizado multitarefa e técnicas de sumarização extrativa, utilizando modelos baseados em BERT para identificar papéis retóricos em sentenças jurídicas e o algoritmo *Maximal Marginal Relevance* (MMR) para selecionar os trechos mais relevantes.

Testado em dados do *Board of Veterans' Appeals*, o modelo apresentou métricas *ROUGE* elevadas. A contribuição foi demonstrar a eficácia de técnicas multitarefa em cenários com dados limitados, melhorando a qualidade de resumos complexos. A limitação é a necessidade de modelos pré-treinados específicos para o domínio jurídico.

No mesmo ano, (ZHANG, 2022) ressaltou a necessidade de técnicas avançadas diante do volume massivo de dados digitais, criticando a ineficiência dos métodos tradicionais para gerenciar informações em larga escala. A contribuição desse trabalho foi enfatizar a importância de modelos avançados, como os baseados em BERT, para lidar com *big data* textual.

3.3.2 Análise textual em mídias sociais

(FLAYEH; HAMODI; ZAKI, 2022), em 2022, abordaram o problema de lidar com o grande volume e variedade de dados em mídias sociais. A metodologia envolveu o uso de PLN e inteligência artificial, com a criação de uma interface em C++ para facilitar a extração de padrões linguísticos.

Os resultados mostraram que técnicas de PLN podem ser eficazes na análise de *tweets* e outras mídias sociais. A contribuição foi fornecer ferramentas para lidar com a heterogeneidade e o alto volume de dados online. A limitação é a necessidade de atualização constante das técnicas para acompanhar a evolução da linguagem nas mídias sociais.

Em 2023, (CHAN; YANG, 2023) enfrentaram o desafio de analisar sentimentos em textos curtos e ruidosos, comuns em plataformas como o *Instagram*. A metodologia combinou LSTM e CNN com *word embeddings* para capturar padrões temporais e semânticos. A coleta de dados foi realizada via *web scraping*, com tradução automática para uniformizar os textos.

Os resultados mostraram uma precisão superior a 79%, evidenciando a eficácia da abordagem. A contribuição foi demonstrar que modelos híbridos são eficazes na análise de sentimentos em redes sociais. A limitação está nas possíveis perdas semânticas devido à tradução automática e à variabilidade linguística.

3.3.3 Detecção de fake news com modelos BERT avançados

(FAROKHIAN; RAFE; VEISI, 2023), em 2023, abordaram o problema de melhorar a detecção de *fake news* utilizando informações do título e do corpo do texto. A metodologia envolveu o MWPBert, que utiliza duas redes BERT em paralelo, processando separadamente o título e o corpo, com o algoritmo *MaxWorth* para selecionar as partes mais relevantes.

Testada em um conjunto de dados de notícias *online*, a arquitetura alcançou uma precisão de 95%. A contribuição foi demonstrar que o uso simultâneo de duas redes BERT melhora a captura de nuances semânticas. A limitação é a maior complexidade computacional.

No mesmo ano, (RAO et al., 2023) desenvolveu uma RNN leve focada na eficiência de memória e processamento em tempo real para a detecção de *fake news*. A metodologia priorizou a redução do consumo de recursos, mantendo alta acurácia.

Os resultados foram positivos, com desempenho satisfatório em bases de dados sobre COVID-19 e outras notícias. A contribuição foi mostrar que modelos leves podem ser eficazes na detecção em tempo real. A limitação é a possível perda de capacidade de capturar padrões complexos.

Em 2024, (RAO et al., 2024) continuaram esse trabalho, enfatizando a importância da eficiência em processamento e memória para a detecção de *fake news* em tempo real. A contribuição foi reforçar a necessidade de modelos otimizados para aplicações práticas.

3.3.4 Automação na geração de taxonomias e classificação de textos

(WAN et al., 2024), em 2024, enfrentaram o problema de criar sistemas de classificação escaláveis sem intervenção humana intensiva. A metodologia envolveu o *Taxonomy and Text classification using Large Language Model* (TNT-LLM), um *framework* que utiliza *Large Language Models* (LLM)s para automação na geração de taxonomias de rótulos e classificação de textos.

Aplicado ao *Bing Copilot*, o método usou aprendizado *zero-shot* para gerar taxonomias refinadas e LLMs como anotadores para expandir o conjunto de treinamento. Os resultados mostraram uma precisão superior a 90%, superando abordagens tradicionais. A contribuição foi demonstrar a escalabilidade e eficiência do modelo. A limitação é a dependência de modelos de linguagem de grande porte, que requerem recursos computacionais significativos.

3.3.5 Conectando trabalhos relacionados com a pesquisa atual

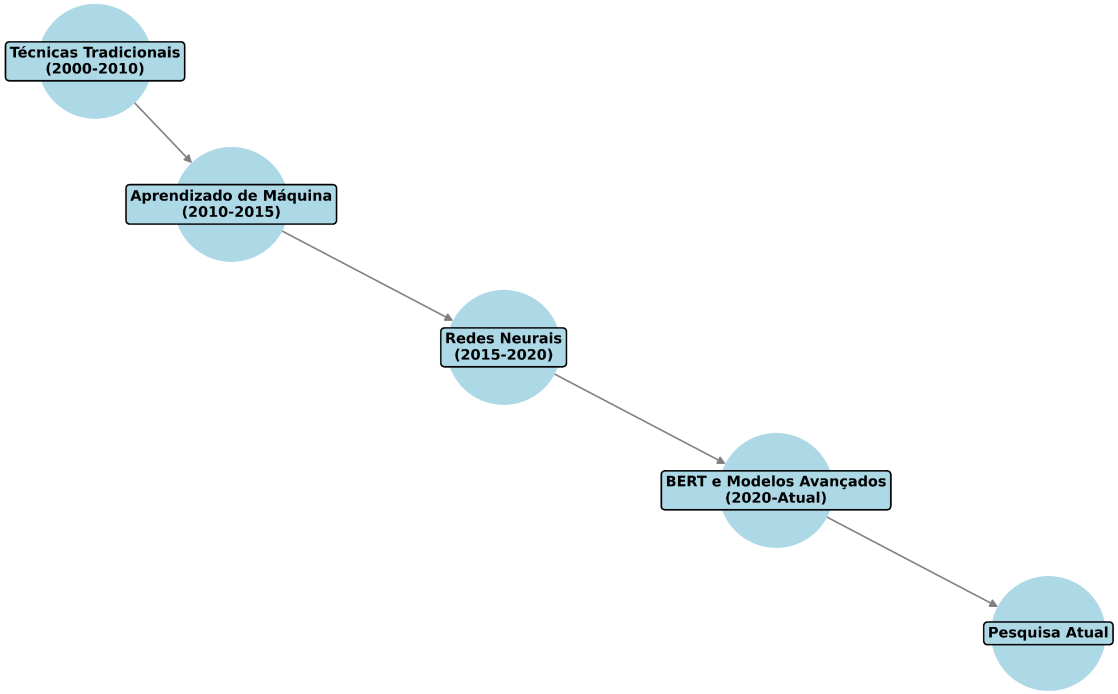
A revisão dos trabalhos evidencia a evolução das técnicas de classificação de texto, desde algoritmos tradicionais até modelos baseados em BERT. Observa-se que métodos tradicionais, embora eficazes em certos contextos, enfrentam limitações ao lidar com a complexidade e o volume dos dados atuais. Modelos de aprendizado profundo e LLMs têm demonstrado maior capacidade de capturar nuances semânticas e contextuais, essenciais para tarefas como detecção de *fake news* e análise de sentimentos.

A presente pesquisa se posiciona nesse contexto, buscando combinar técnicas de processamento de linguagem natural com aprendizado profundo para construir uma base sólida na classificação de textos. Inspiramo-nos em trabalhos como o de (FAROKHIAN; RAFE; VEISI, 2023), que utilizam múltiplas redes BERT para melhorar a detecção de *fake news*, e em abordagens que enfatizam a eficiência computacional, como a de (RAO et al., 2023).

Ao reconhecer as limitações identificadas, como a necessidade de grandes volumes de dados rotulados e a alta demanda computacional, nossa pesquisa propõe soluções que equilibram precisão e eficiência. Exploramos técnicas de pré-processamento textual para otimizar o desempenho dos modelos de aprendizado profundo, visando aplicabilidade prática em cenários com recursos limitados.

Na Figura 1, está sendo apresentado um fluxograma que ilustra a evolução das técnicas de classificação de texto dos trabalhos revisados nesta seção. O fluxograma mostra a progressão cronológica das técnicas de classificação de texto, desde as técnicas tradicionais até a sua pesquisa atual, destacando as principais etapas evolutivas.

Figura 1 – Evolução das técnicas de classificação de texto



Fonte: Elaborada pelo autor.

3.3.6 Síntese e relevância para o estudo

Os trabalhos revisados demonstram que a classificação de textos é uma área dinâmica, com avanços contínuos impulsionados por desafios emergentes, como a disseminação de *fake news* e a necessidade de processar grandes volumes de dados em tempo real. As técnicas tradicionais oferecem fundamentos importantes, mas os modelos baseados em aprendizado profundo, especialmente aqueles que utilizam BERT e outras arquiteturas avançadas, têm se mostrado superiores em precisão e capacidade de generalização.

A relevância deste estudo está em sua contribuição para o aprimoramento dessas técnicas, ao propor abordagens que consideram as limitações práticas identificadas na literatura. Ao combinar técnicas de pré-processamento eficientes com modelos de aprendizado profundo otimizados, buscamos desenvolver soluções que sejam não apenas precisas, mas também viáveis em termos de recursos computacionais.

Dessa forma, este trabalho não apenas avança o estado da arte na classificação de textos, mas também oferece *insights* valiosos para aplicações práticas, contribuindo para o desenvolvimento de sistemas mais robustos e eficientes em um cenário de informações cada vez mais complexo e volumoso.

Em resumo e para concluir as análises dos trabalhos nesta seção, a Tabela 1 apresenta um comparativo entre as diferentes abordagens utilizadas pelos trabalhos relacionados discutidos neste capítulo.

Tabela 1 – Comparação das abordagens dos trabalhos relacionados.

Autores e Ano	**Técnica**	**Base de Dados**	**Acurácia/F1-score**	**Limitações**
Zhang et al. (2009)	Fuzzy k-NN (Aprendizado de Máquina)	Documentos da *web*	F1-score de 76,38%	Menor velocidade de classificação comparado ao k-NN
Pivetta et al. (2013)	Naive Bayes com Janela Deslizante e n-gramas	200 Boletins Internos de 64 militares	F1-score de 76,7%	Não especificado
Ahmed et al. (2017)	N-grams e aprendizado de máquina NB, RL)	25.200 notícias reais e falsas	Acurácia de 92%	Não especificado
Wang (2018)	DRNN (Rede Neural)	7 bases incluindo 20 *Newsgroups* e *Reuters*	F1-score acima de 90%	Resolver limitações das RNNs tradicionais
Miao et al. (2018)	KNN, NB, SVM (aprendizado de máquina)	Base de notícias da Universidade Fudan	SVM atingiu 95% de precisão e F1-score	Importância da modelagem independente da base
Cherif et al. (2021)	Classificação por limiares e redução de dimensionalidade	BBC *News* e *Reuters*	F1-score acima de 95%	Não especificado
Chan e Yang (2023)	LSTM e CNN com *Word Embeddings* (redes neurais)	Postagens do Instagram via *web scraping*	Acurácia acima de 79%	Eficaz para textos curtos e ruidosos
Farokhian et al. (2023)	MWPBert com duas redes BERT em paralelo (rede neural)	Conjunto de notícias online	Acurácia de 95%	Não especificado
Rao et al. (2023)	RNN leve para detecção de *fake news* (rede neural)	Notícias sobre COVID-19 e contemporâneas	Acurácia de 94%	Importância do pré-processamento em cenários dinâmicos
Wan et al. (2024)	TnT-LLM com LLMs e aprendizado zero-shot (rede neural)	*Bing Copilot* (busca conversacional)	Acurácia acima de 90%	Escalabilidade onde a intervenção humana é limitada

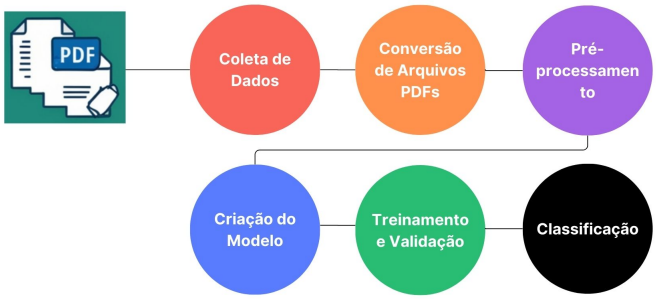
Fonte: Elaborada pelo autor.

Capítulo 4

Materiais e métodos

A metodologia adotada neste trabalho caracteriza-se pela natureza de pesquisa aplicada, com abordagem quantitativa e objetivo de pesquisa exploratória. Para alcançar os objetivos propostos, foram explorada diversas técnicas, discutidas na seção de trabalhos relacionados, que englobam tanto técnicas tradicionais já consolidadas quanto métodos mais recentes, seguindo procedimentos bibliográficos conforme a literatura. Inicialmente, considerou-se utilizar algoritmos tradicionais de aprendizado de máquina, como o Naive Bayes, para a construção de um classificador binário de documentos textuais. No entanto, em virtude das técnicas mais recentes e eficazes na classificação de textos, optou-se por focar na utilização de modelos de deep learning e na aplicação de técnicas avançadas de PLN para o pré-processamento dos dados. A Figura 2 apresenta uma visão geral das etapas da proposta nesta dissertação.

Figura 2 – Fluxo das etapas da proposta



Fonte: Elaborada pelo autor.

Conforme ilustrado na Figura 2, nossa metodologia segue uma sequência lógica de etapas, desde a coleta de dados até a classificação. Durante a fase de pré-processamento, foram aplicadas diversas técnicas de PLN para preparar os dados. Realizou-se a normalização, processo que converte as palavras para uma forma padrão, como a transformação

de caracteres maiúsculos em minúsculos. Em seguida, efetuou-se a tokenização utilizando a biblioteca *SpaCy*, que divide o texto em unidades léxicas. Posteriormente, foram removidas a pontuação e os caracteres especiais para limpar o texto. Também foram eliminadas as *stop words*, utilizando uma lista de palavras comuns que não carregam significado importante, como artigos e preposições. Além disso, aplicou-se a técnica de lematização, também com a biblioteca *SpaCy*, para reduzir as palavras às suas formas base, removendo sufixos e prefixos. Outros procedimentos de manipulação e limpeza de dados foram realizados utilizando a biblioteca Pandas.

Diferentes tipos de normalização podem ser adotados, dependendo do problema a ser resolvido e da abordagem escolhida para tratar de maneira mais eficiente o processamento e padronização textual. Esta etapa de pré-processamento é crucial para garantir que o texto esteja em uma forma normalizada para a criação do conjunto de dados e, consequentemente, para a realização da classificação. A combinação das abordagens de PLN, tanto tradicionais quanto recentes, foi essencial para alcançar os resultados esperados, representando uma das principais contribuições deste trabalho. Para um melhor entendimento de todas as etapas, uma imagem do fluxo de trabalho é apresentada na Figura 2, seguida das descrições detalhadas de cada uma das etapas propostas.

4.1 Base de dados

A construção da base de dados deste trabalho envolveu etapas fundamentais para garantir a limpeza, adequação e a qualidade dos dados ao treinamento dos modelos. Como os dados foram obtidos a partir de fontes não estruturadas, como textos livres em arquivos no formato PDF extraídos de páginas *web*, foi necessário realizar um tratamento cuidadoso para transformá-los em um *dataset* organizado. Após esse processamento, o *dataset* foi exportado no formato *Comma-Separated Values* (CSV) para facilitar sua manipulação e depois utilizado em três modelos diferentes de RN. A base de dados contendo 2738 registros e duas classes, foi utilizada para servir de entrada para os modelos implementados durantes as etapas de treinamento, validação e teste. Esse *dataset* foi dividido em três partes: 70% para o treinamento do modelo, 15% para validação, com foco na otimização de hiperparâmetros, e 15% para teste, destinado a avaliar o desempenho do modelo em dados não vistos.

Além do processo de divisão, observou-se que as classes não estavam balanceadas, conforme pode ser observado na Tabela 2. Em próximos experimentos, pode ser testado a realização do balanceamento das classes para verificar se esse processo pode garantir que os modelos tenham melhores resultados se os dados estivessem balanceados entre as duas classes e assim evitar qualquer vieses nos modelos treinados. Na realização da análise exploratória dos dados, verificou-se como ficou a distribuição dos registros entre as classes, conforme ilustrado na Tabela 2. Adicionalmente, técnicas de pré-processamento

como remoção de termos duplicados foram aplicadas para aprimorar a qualidade do *dataset*. Esse cuidado na preparação dos dados assegurou que os modelos de redes neurais recebessem informações consistentes e representativas, aumentando a confiabilidade dos resultados obtidos nas etapas de validação e teste.

A Tabela 2 apresenta como as classes ficaram distribuídas no *dataset*:

Tabela 2 – Distribuição das classes

Classe	**Quantidade de Registros**
0	1521
1	1217
Total	**2738**

Fonte: Elaborada pelo autor.

4.2 Etapas da proposta

4.2.1 Coleta dos dados

Nesta fase, foi utilizado recursos tecnológicos, como a linguagem de programação *Python* para o desenvolvimento de um *crawler*, ou seja, foi utilizado a técnica de *web scraping* que é a raspagem de dados na *web*. Foi necessário desenvolver o *crawler* para realizar de forma autmatizada o acesso ao portal do diário oficial estadual com dados abertos e baixar os arquivos formato PDF que são portarias publicados diariamente no portal. Os documentos coletados e utilizados no experimentos, foram do ano de 2023 e as informações textuais extraídas desses documentos foram utilizadas nas etapas subsequentes, principalmente na fase de pré-processamento dos dados e na construção do *dataset*. Além disso, esse procedimento de pré-processamento é necessário porque nas fases de treinamento, validação e teste do classificador o custo computacional é alto e o conjunto de dados precisa estar trato e sem inconsistências.

Para automatizar a coleta de arquivos PDF a partir do portal do Diário Oficial de acesso aberto da Defensoria Pública do Tocantins, desenvolveu-se um *crawler* utilizando a linguagem de programação *Python*. O objetivo foi percorrer sistematicamente um intervalo de páginas *web*, extrair os *links* e baixar os documentos em formato PDF para um diretório local. Para aprimorar a organização e a manutenibilidade do código, aplicaram-se os princípios da Programação Orientada a Objetos (POO), encapsulando as funcionalidades do *crawler* dentro de uma estrutura de classes.

A funcionalidade central do *crawler* foi encapsulada em uma classe denominada *PDFCrawler*, que incluiu métodos para inicialização de parâmetros, navegação pelas páginas *web*, extração de *links* de PDFs e download dos documentos. O método de inicialização configurou as *URL*s base, o intervalo de páginas a serem processadas e o diretório de saída para os arquivos baixados. O método de navegação percorreu o intervalo especificado, en-

viando requisições *Hypertext Transfer Protocol* (HTTP) para cada página, analisando o conteúdo HTML com a biblioteca *BeautifulSoup* e extraindo os *links* das páginas *web* no diário oficial com base em atributos HTML específicos.

Para o *download* dos arquivos PDF, implementou-se um método que iterou sobre os *links* extraídos, verificando a existência prévia dos arquivos para evitar duplicações, e realizou requisições HTTP para obter o conteúdo dos PDFs. O tratamento de exceções foi incorporado para gerenciar potenciais erros durante o processo de *download*, assegurando a eficiência e a confiabilidade do programa. O design modular do *crawler*, foi facilitado pelos princípios da POO, permitiu melhorar a legibilidade do código, facilitar a manutenção e oferecer flexibilidade para ajustar parâmetros, como o intervalo de páginas para a coleta de dados. Com os dados coletados e armazenados, avançamos para a etapa da conversão dos arquivos PDF para texto, essencial para preparar os textos para o pré-processamento.

A Figura 13 na seção do Apêndice A deste trabalho apresenta o código completo implementado e utilizado para realizar a coleta de dados usando as técnicas mencionadas acima.

4.2.2 Conversão dos arquivos PDF

Nesta etapa, os documentos em formato PDF foram convertidos para um formato textual, facilitando as etapas seguintes do projeto. A biblioteca *open source PyPDF2*, desenvolvida em *Python*, foi utilizada para realizar essa conversão dos dados brutos, e as informações textuais resultantes foram armazenadas em uma base de dados no formato CSV, ou seja, foi criado um *dataset* com todos dados coletados no passo anterior.

Para converter os arquivos PDF em um único arquivo de texto e, em seguida, criar e exportar um *dataset* estruturado, desenvolveu-se um programa que automatizou a extração de texto dos PDFs, a rotulação de parágrafos com base em palavras-chave específicas e a exportação dos dados em formato CSV. Inicialmente, o programa percorreu todos os arquivos PDF presentes em um diretório determinado, utilizando a biblioteca os para manipulação de arquivos e *PyPDF2* para leitura dos PDFs. Cada arquivo PDF foi aberto e processado página por página, extraindo-se o texto contido em cada uma delas. O texto extraído foi concatenado em uma única *string*, mantendo a formatação original dos parágrafos ao inserir quebras de linha duplas entre eles.

Após a extração completa do texto de cada PDF, o conteúdo foi dividido em parágrafos individuais, segmentando a *string* nas quebras de linha duplas. Cada parágrafo foi então analisado pela função de rotulação, que verificou a presença de palavras-chave específicas, tais como "DESIGNAR", "REVOGAR", "PRORROGAR", "ALTERAR"e "SUSPENDER". Se uma dessas palavras-chave estivesse presente no parágrafo, este recebia um rótulo com o número (1); caso contrário, era atribuído um rótulo com o número (0). Essa etapa de rotulação inicial, foi essencial para categorizar os parágrafos conforme sua rele-

vância para os critérios estabelecidos, permitindo uma organização adequada dos dados para análises posteriores.

Por fim, os parágrafos extraídos e seus respectivos rótulos foram armazenados em uma estrutura de dados adequada, geralmente uma lista de dicionários. Utilizando a biblioteca pandas, essa lista foi convertida em um *DataFrame*, que estruturou os dados em formato tabular com colunas para o texto dos parágrafos e seus rótulos correspondentes. Este *DataFrame* foi então exportado para um arquivo CSV, facilitando a persistência dos dados e sua utilização em futuros processos, como análises estatísticas ou treinamentos de modelos de aprendizado de profundo. A lógica do programa incorporou conceitos fundamentais de programação, como manipulação de arquivos, iteração, funções e estruturas de dados, além do uso de bibliotecas especializadas para processamento de PDFs e manipulação de dados, o que permitiu uma conversão eficiente e automatizada dos dados de PDF para um formato estruturado e analisável.

A Figura 14 na seção do Apêndice A deste trabalho apresenta o código completo implementado e utilizado para realizar a conversão dos dados extraídos dos arquivos PDF para o arquivo de texto e posteriormente criar o *dataset* usando as bibliotecas apresentadas neste trabalho.

4.2.2.1 Limpeza dos dados

Inicialmente, para compreender a dimensão e a estrutura do *dataset*, utilizou-se o comando *data.shape*, que retorna a quantidade de linhas e colunas presentes. Esse passo é fundamental para ter uma visão geral do tamanho dos dados com os quais se está trabalhando. Em seguida, para identificar valores ausentes nas colunas, aplicou-se *data.isnull ().sum()*, que fornece o número de valores nulos *NaN* em cada coluna, permitindo detectar quais atributos possuem dados faltantes que necessitam de tratamento.

Para lidar com os valores ausentes especificamente na coluna 'texto', considerou-se a remoção das linhas onde esses valores estavam presentes. Isso foi feito com o comando *data.dropna(subset=['texto'], inplace=True)*, que exclui do *dataset* todas as linhas em que a coluna 'texto' possui valores nulos. Alternativamente, para analisar as linhas com qualquer valor ausente sem removê-las imediatamente, utilizou-se *nan-values = data[data.isna().any(axis=1)]*, isolando as entradas incompletas para uma possível inspeção ou tratamento posterior. O total de valores ausentes em todo o *dataset* foi obtido com *data.isnull().sum().sum()*, oferecendo uma visão global da incompletude dos dados.

No processo de limpeza textual, foi necessário remover caracteres indesejados que poderiam interferir nas análises subsequentes. Para eliminar quebras de linha, o comando *data['texto'] = data['texto'].str.replace('\n', ' ')* substituiu os caracteres *'\n'* por espaços em branco, tornando o texto contínuo e facilitando sua manipulação. Da mesma forma, caracteres de retorno *('\r')*, que representam o retorno do cursor ao início da linha, foram removidos usando *data['texto'] = data['texto'].str.replace('\n', ' ')*. Essas substituições

padronizam o formato do texto, eliminando inconsistências causadas por diferentes formas de quebra de linha.

A padronização do espaçamento dentro dos textos também foi essencial. Para remover espaços em branco no início e no fim das strings da coluna 'texto', aplicou-se *data['texto'] = data['texto'].str.strip()*. Isso garante que não haja espaços extras que possam afetar a qualidade da análise textual. Além disso, para substituir múltiplos espaços em branco consecutivos por um único espaço, utilizou-se *data['texto'] = data['texto'].str.replace(' +', ' ', regex=True)*. Esse comando, com o auxílio de expressões regulares, normaliza o espaçamento entre as palavras, corrigindo possíveis erros de digitação ou formatação.

No que diz respeito aos valores ausentes após as etapas anteriores, optou-se por preenchê-los ao invés de removê-los, preservando assim o máximo de informações possível. O comando *data.fillna(1, inplace=True)* substituiu todos os valores *NaN* restantes pelo número 1, que pode representar um valor padrão ou categoria específica conforme o contexto do *dataset*. Para assegurar que não restassem valores ausentes, novas verificações foram feitas com *data.isnull().sum()* e *data.isnull().sum().sum()*, confirmando a completude dos dados.

Por fim, ajustou-se a consistência dos valores na coluna 'rotulo'. Com o comando *data['rotulo'] = data['rotulo'].replace(1, 0)*, todos os valores iguais a 1 foram substituídos por 0. Essa operação pode ter sido necessária para unificar a representação das classes ou corrigir possíveis inconsistências nos rótulos, garantindo que a variável alvo estivesse corretamente configurada para as análises ou modelos preditivos que seriam aplicados posteriormente.

4.2.3 Pré-processamento dos dados

Para que os modelos de *deep learning* realizassem a tarefa de classificação, os dados textuais passaram por um processo de pré-processamento antes de serem utilizados no treinamento, validação e teste, a fim de adequá-los ao formato exigido pelo modelo e garantir sua eficiência. Esse pré-processamento incluiu técnicas de processamento de linguagem natural, como lematização, remoção de pontuações, eliminação de *stop words*, tokenização e vetorização com o objetivo de estruturar e normalizar os textos para otimizar o desempenho dos modelos. Para a implementação dessas etapas, foram utilizadas as bibliotecas *spaCy* e Pandas, ambas desenvolvidas em Python e escolhidas por sua robustez e eficiência na manipulação de dados textuais. As técnicas de pré-processamento foram selecionadas para otimizar a qualidade dos dados e melhorar o desempenho dos modelos. A lematização foi escolhida para reduzir as palavras às suas formas base, preservando o contexto semântico, o que é crucial em documentos oficiais. A remoção de *stop words* e pontuações foi realizada para eliminar ruídos e focar nos termos relevantes para a classificação. As Figuras 3 e 4 apresentam o antes e o depois do pré-processamento dos dados. Na Figura 3 está sendo apresentado o *dataset* antes do pré-processamento dos dados:

Figura 3 – Base de dados completa antes do pré-processamento.

	texto	rotulo
0	Edição \nNª 354 - Publicada em 31/10/2022\nCON...	0
1	Art. 2º.\n Fica criado o artigo 5º-A da Resolu...	0
2	CONSIDERANDO \no histórico envolvimento nacion...	0
3	Documento assinado eletronicamente por \nEstel...	0
4	(CNPJ 37.552.014/0001-28), vencedora do item 2...	0
...	...	...
5261	Assinatura de Publicação: xidag-kogec-zacyl-ny...	1
5262	RESOLVE: Art. 1º DESIGNAR a Defensora Pública ...	0
5263	O\n \nSUBDEFENSOR PÚBLICO-GERAL DO ESTADO DO T...	1
5264	RESOLVE: Art. 1º DESIGNAR a Defensora Pública ...	0
5265	DEFENSORIA PÚBLICA DO TOCANTINS, QUARTA-FEIRA,...	0

5266 rows × 2 columns

Fonte: Elaborada pelo autor.

Já na Figura 4 está sendo apresentado o *dataset* final depois da limpeza e do pré-processamento dos dados:

Figura 4 – Base de dados final depois do pré-processamento.

	texto	rotulo
0	edicao nª 354 publicar 31/10/2022 superior res...	0
1	art. 2º. ficar criar artigo 5º-a resolucao-csd...	0
2	considerando historico envolvimento nacional e...	0
3	cnpj 37.552.014/ 0001-28 vencedor item 21 tota...	0
4	lucindo atribuicao 1ª defensoria publica famil...	0
...	...	...
2733	resolve art. 1º designar defensor publico 1ª c...	1
2734	resolve art.1º alterar razao extremo necessida...	1
2735	resolve art. 1º designar defensora publica 1ª ...	1
2736	resolve art. 1º designar defensora publica 1ª ...	1
2737	resolve art. 1º designar defensora publica 1ª ...	1

2738 rows × 2 columns

Fonte: Elaborada pelo autor.

Abaixo, detalham-se as principais abordagens utilizadas

4.2.3.1 Conversão para minúsculas

A conversão de todo o texto para minúsculas é uma técnica essencial no pré-processamento. Como a maioria dos modelos considera palavras maiúsculas e minúsculas como diferentes, essa padronização permite reduzir a dimensionalidade e evitar duplicação de termos semanticamente iguais. Por exemplo, palavras como (Brasil) e (brasil) são tratadas como idênticas após essa conversão.

4.2.3.2 Remoção de pontuações

A pontuação presente nos textos, como pontos finais, vírgulas e pontos de interrogação, é irrelevante para a maioria das tarefas de PLN. Assim, foi realizada a remoção desses elementos, a fim de eliminar ruído e evitar que a presença de símbolos interfira no desempenho dos modelos.

4.2.3.3 Eliminação de stop words

Stop words são palavras de alta frequência que geralmente não agregam significado relevante ao conteúdo do texto, como artigos e preposições (e.g., “o”, “a”, “de”, “para”). A eliminação desses termos contribui para uma análise mais focada no conteúdo informativo dos dados textuais. Nos experimentos realizados, a lista de *stop words* foi fornecida pelo *spaCy* para o idioma português.

4.2.3.4 Lematização com spaCy

A lematização consiste em reduzir as palavras à sua forma base (lema). Ao contrário da simples remoção de sufixos (*stemming*), a lematização preserva a integridade gramatical da palavra. Por exemplo, a palavra (correndo) é lematizada para (correr). A técnica foi implementada utilizando a biblioteca *spaCy*, que oferece suporte robusto para a língua portuguesa e facilita a normalização linguística, aumentando a eficácia dos modelos de PLN.

4.2.3.5 Tokenização

A tokenização divide textos em unidades menores, chamadas *tokens*, essenciais para o processamento textual. Para o modelo BERT, foi utilizado seu próprio *tokenizer*, que fragmenta palavras em subpalavras e inclui tokens especiais, como [CLS] e [SEP], para otimizar a classificação e preservação do contexto. Já para os modelos RNN e LSTM, utilizou-se o *tokenizer* da biblioteca *spaCy*. Essa abordagem permitiu a preparação dos textos em *tokens*, seguindo uma lógica específica para transformar o conteúdo em tensores compatíveis com os modelos de redes neurais, utilizando a biblioteca *PyTorch*.

4.2.3.6 Vetorização

A vetorização transforma *tokens* em vetores numéricos que os modelos de aprendizado profundo conseguem interpretar. No BERT, a vetorização é realizada pelo próprio modelo, utilizando contextual *embeddings* que capturam nuances semânticas e sintáticas dos textos. Para os modelos RNN e LSTM, foi criada uma função personalizada que converte os *tokens* em tensores por meio da biblioteca *SpaCy* e linguagem de programação Python, garantindo que os dados estejam no formato esperado pelas redes neurais e otimizando seu desempenho durante o treinamento e a classificação.

4.2.3.7 Word Embedding

Word embedding é uma técnica que transforma palavras em vetores densos de baixa dimensionalidade, onde termos semanticamente semelhantes apresentam representações próximas no espaço vetorial. Essa abordagem permite que os modelos de redes neurais capturem as relações semânticas entre palavras e frases. No caso das redes RNN e LSTM, os *embeddings* foram gerados para garantir que as informações contextuais fossem preservadas ao longo das sequências textuais. Essa técnica é fundamental para melhorar a capacidade dos modelos de lidar com padrões complexos no texto durante o treinamento e a inferência.

4.2.3.8 GloVe

GloVe Global Vectors (GV) para *Word Representation* (WR) é uma técnica de *word embedding* baseada na coocorrência global de palavras em um grande corpus. Ela cria vetores que capturam tanto a semântica quanto a frequência das palavras, produzindo representações eficazes para modelos de processamento de linguagem natural. Nos experimentos realizados com RNN e LSTM, utilizou-se o *GloVe* para treinar e inicializar os *embeddings*, fornecendo ao modelo informações semânticas e acelerando o processo de treinamento ao partir de vetores pré-treinados.

4.2.4 Criação do modelo

Nesta fase, foi criado e testado diversos modelos de classificadores baseados em redes neurais utilizando redes com arquiteturas dos modelos de *deep learning*. Foram realizados experimentos, incluindo testes de hiperparâmetros, para analisar e validar a performance dos modelos. Os dados utilizados para treinamento, validação e teste foram provenientes das etapas anteriores. Optamos por modelos RNN e LSTM devido à sua capacidade de capturar relações sequenciais em textos, o que é importante em documentos com linguagem formal e estrutura específica. O BERT foi selecionado por ser um modelo de estado da arte que utiliza mecanismos de atenção, permitindo uma compreensão mais profunda do contexto.

4.2.4.1 Hiperparâmetros

Os hiperparâmetros utilizados na criação e treinamento dos modelos RNN, LSTM e BERT foram ajustados para garantir a melhor performance nas tarefas de classificação. Estes parâmetros controlam aspectos essenciais, como a arquitetura das redes, o processo de treinamento e a generalização dos modelos. A seguir, detalham-se os hiperparâmetros aplicados:

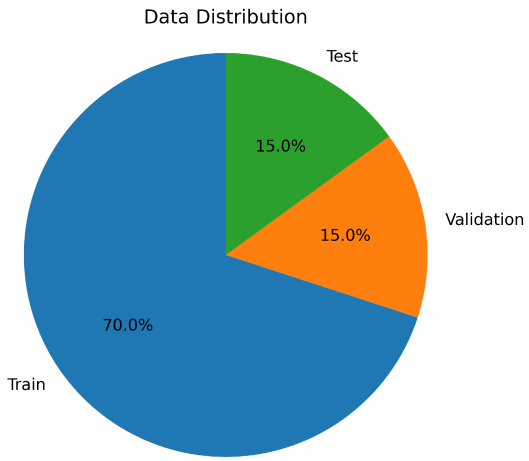
1. $max_length = 100$: Define o tamanho máximo das sequências de entrada, padronizando o comprimento dos textos.
2. $embedding_dim = 100$: Tamanho da dimensão dos vetores de *embedding* para representar cada palavra.
3. $hidden_dim = 128$: Número de neurônios nas camadas ocultas das redes RNN e LSTM.
4. $output_dim = 1$: Saída binária, indicando a classificação em uma das duas categorias possíveis.
5. $batch_size = 32$: Número de amostras processadas em cada lote de treinamento.
6. $num_epochs = 10$: Quantidade de ciclos completos de treinamento sobre todo o conjunto de dados.
7. $learning_rate = 0.001$: Taxa de aprendizado para controlar o ajuste dos pesos durante o treinamento.
8. $dropout_prob = 0.5$: Probabilidade de exclusão de neurônios para evitar *overfitting*.
9. $patience = 3$: Quantidade de épocas sem melhoria na métrica de validação para aplicar *early stopping*.
10. $criterion =$ nn.BCELoss(): Função de perda utilizada para problemas de classificação binária.
$$BCELoss = -(y * logp + (1 - y) * log(1 - p)) \quad (1)$$
11. $optimizer =$ optim.Adam: Otimizador utilizado para ajuste dos pesos dos modelos durante o treinamento.

Esses hiperparâmetros foram cuidadosamente ajustados para os três modelos, garantindo que tanto as redes RNN e LSTM quanto o BERT operassem com eficiência, equilibrando desempenho e capacidade de generalização.

4.2.5 Treinamento e validação

Nesta etapa, o treinamento do modelo foi realizado utilizando dados da fase de pré-processamento dos dados para aprender a relação entre entradas e saídas, onde o modelo fez previsões e ajustou seus pesos através de *backpropagation* para minimizar uma função de perda. Durante a validação, que ocorreu simultaneamente ao treinamento, o modelo foi testado com dados não utilizados anteriormente para avaliar sua capacidade de generalização, monitorando métricas como acurácia, precisão, *recall* e F1-*score*. A validação também foi crucial para o ajuste de hiperparâmetros e após a validação, um conjunto de teste foi utilizado para uma avaliação final, confirmando a eficácia do modelo em condições reais. Este ciclo de treinamento e validação foi essencial para desenvolver um modelo robusto e eficaz. Na Figura 5 está sendo apresentada como foi realizado a divisão dos dados para estas etapas.

Figura 5 – Distribuição dos dados



Fonte: Elaborada pelo autor.

4.2.6 Classificação

Na etapa de classificação, foi o momento de medir a eficácia dos classificadores de textos baseados em redes neurais. Foram realizados testes de classificação com o objetivo de categorizar os textos em duas classes distintas por se tratar de uma classificação binária. O processo de classificar foi possível devido às etapas anteriores essenciais que transformaram os textos, isto é, os dados brutos em dados no formato em que os classificadores necessitam para realizar as previsões classificatórias.

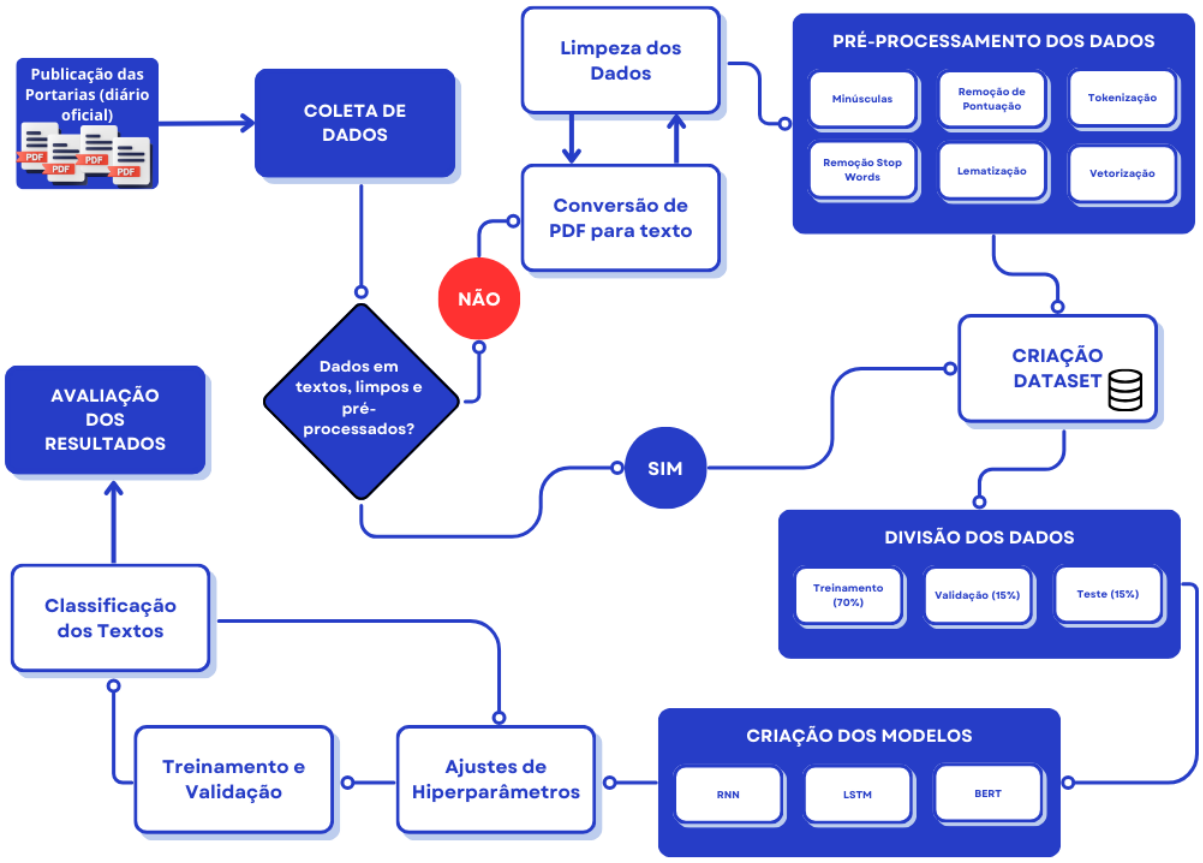
4.3 Considerações éticas e legais

Todos os dados utilizados foram obtidos de fontes públicas e a coleta de dados respeitou as políticas de acesso e uso dos dados disponível no portal do Diário Oficial da Defensoria Pública do Estado do Tocantins. Em relação à privacidade, nenhum dado pessoal sensível foi coletado ou utilizado.

4.4 Arquitetura da proposta

Para finalizar esta seção do trabalho e fazer um resumo de todas as etapas mencionadas e detalhadas acima, a Figura 6 está ilustrando a arquitetura da proposta, nesse caso representado por diagrama:

Figura 6 – Arquitetura da proposta



Fonte: Elaborada pelo autor.

Capítulo 5

Experimentos e Análise dos Resultados

Neste capítulo, são apresentados os experimentos realizados e a análise dos resultados obtidos ao longo da pesquisa. O objetivo principal é avaliar o desempenho de modelos de aprendizado profundo, especificamente RNN, LSTM e BERT na classificação de textos jurídicos. Para o desenvolvimento dos experimentos, foi utilizado a linguagem *Python* juntamente com as bibliotecas *PyTorch* e Pandas para a implementação dos modelos. Além disso, o ambiente *Google Colab* foi empregado para aproveitar os recursos de processamento em GPU, acelerando o treinamento das redes neurais.

Os dados foram submetidos a um rigoroso processo de pré-processamento, aplicando técnicas avançadas de PLN conforme descrito na metodologia. Em seguida, o conjunto de dados foi dividido em três partes: treinamento, validação e teste, nas proporções de 70%, 15% e 15%, respectivamente. Essa divisão permite avaliar a capacidade de generalização dos modelos e evitar o *overfitting*.

Os modelos foram treinados e ajustados utilizando o otimizador *Adam*, configurados com hiperparâmetros específicos para cada arquitetura, visando alcançar o melhor desempenho possível. As métricas de avaliação utilizadas incluem acurácia, precisão, *recall* e F1-*score*, fornecendo uma análise da eficácia de cada modelo na tarefa proposta.

Neste capítulo, detalhamos o processo de avaliação dos modelos, os parâmetros definidos, os resultados obtidos em cada experimento e uma discussão comparativa entre as arquiteturas testadas. A análise dos resultados permite validar as hipóteses formuladas e verificar se os objetivos estabelecidos foram atingidos, contribuindo para o avanço na classificação automatizada de textos jurídicos em instituições públicas.

5.1 Métricas de Avaliação

Para validar a hipótese desta pesquisa, foram utilizados métodos que incluem medidas de avaliação, conjunto de parâmetros, bases de dados e comparação com trabalhos relacionados. Os algoritmos de aprendizado profundo foram treinados e avaliados com base nas métricas de desempenho estabelecidas.

5.1.1 Acurácia

A acurácia é uma métrica utilizada para avaliar modelos de classificação, indicando a fração de previsões corretas realizadas pelo modelo. Formalmente, a acurácia é definida como a soma dos Verdadeiros Positivos (VP) e Verdadeiros Negativos (VN) dividida pela soma de VP, VN, Falsos Positivos (FP) e Falsos Negativos (FN) (FERREIRA, 2019). A Equação 2 apresenta a fórmula para calcular a acurácia:

$$Acurácia = \frac{TP + TN}{TP + FP + FN + TN} \tag{2}$$

5.1.2 Precisão

A precisão representa a proporção de previsões positivas que foram corretas. Um modelo que não produz falsos positivos possui uma precisão igual a 1. Formalmente, a precisão é definida como a quantidade de VP dividida pela soma de VP e FP (FERREIRA, 2019). A Equação 3 apresenta a fórmula para calcular a precisão:

$$Precisão = \frac{TP}{TP + FP} \tag{3}$$

5.1.3 Recall

O *recall* indica a proporção de verdadeiros positivos que foram identificados corretamente. Um modelo que não produz falsos negativos tem um *recall* igual a 1. Formalmente, o *recall* é definido como a quantidade de VP dividida pela soma de VP e FN (FERREIRA, 2019). A Equação 4 apresenta a fórmula para calcular o *recall*:

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

5.1.4 F1-score

O F1-*Score* é a média harmônica entre a precisão e o *recall*. Ele é especialmente útil quando há um *trade-off* entre precisão e *recall*, proporcionando uma métrica única para avaliar o desempenho. O F1-*Score* é particularmente relevante em situações com classes desbalanceadas. A Equação 5 apresenta como é realizado o cálculo da métrica F1-*Score*:

$$F1 - Score = \frac{2 * Precisão * Recall}{Precisão + Recall} \tag{5}$$

5.1.5 Matriz de confusão

Uma das formas mais diretas de avaliar o desempenho de um modelo de classificação é por meio da visualização e análise da Matriz de Confusão (MC). Dessa matriz, é possível

extrair as métricas mencionadas anteriormente, que são amplamente utilizadas para medir o desempenho dos sistemas de aprendizagem (FERREIRA, 2019).

Tabela 3 – Matriz de confusão

	Previsão Positiva	**Previsão Negativa**
Classe Real Positiva	Verdadeiro Positivo (VP)	Falso Negativo (FN)
Classe Real Negativa	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Fonte: Elaborada pelo autor.

5.1.6 Média

A média é uma medida estatística que descreve o valor central de um conjunto de dados, sendo calculada pela soma de todos os valores dividida pelo número total de observações. Em problemas de análise de desempenho de modelos de redes neurais, a média fornece um indicativo representativo do resultado esperado, facilitando a comparação entre diferentes soluções e auxiliando na identificação do comportamento geral de cada abordagem (GRUS, 2016).

$$Média = \frac{M1 + M2 + M3}{3} \tag{6}$$

5.1.7 Desvio Padrão

Já o desvio padrão quantifica a dispersão dos dados em torno da média, refletindo o grau de variação de cada métrica de desempenho. Quanto maior o seu valor, mais heterogênea é a distribuição dos resultados; enquanto valores menores apontam para resultados mais consistentes. Assim, a análise conjunta de média e desvio padrão oferece uma visão do comportamento dos modelos e contribui para uma avaliação mais sólida e confiável (GRUS, 2016).

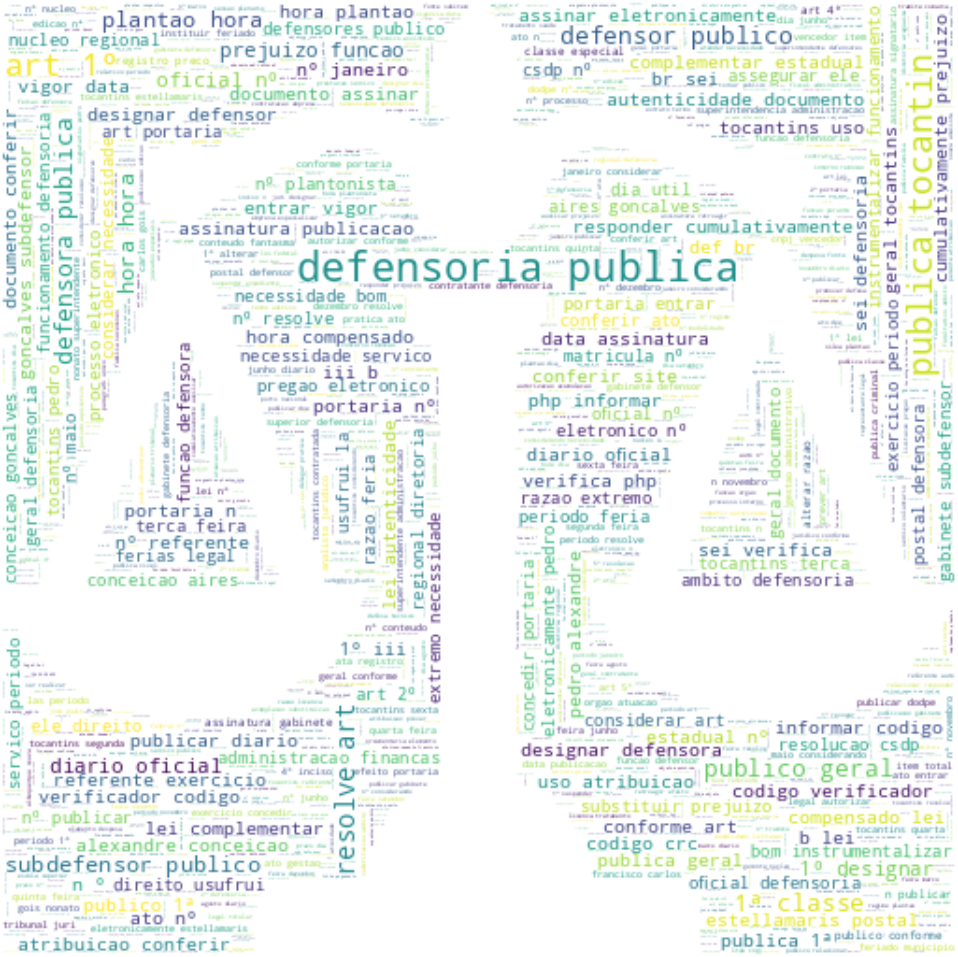
$$\sigma = \sqrt{\frac{\Sigma_{i=1}^{3}(x_i - \bar{x})^2}{3}} \tag{7}$$

5.2 Experimentos

Após os processos de limpeza, rotulação e pré-processamento dos dados, foi gerada uma imagem para representar os termos mais frequentes no *dataset*. A Figura 7 apresenta uma nuvem de palavras que destaca termos e conceitos associados à Defensoria Pública, especialmente em um contexto de regulamentações e procedimentos administrativos relacionados às atribuições de defensores e defensoras públicas por meio de publicações de portarias. As palavras mais evidentes incluem "defensoria pública,tocantins,plantão,documento,oficial,lei complementar"e "assinatura eletronicamente,"sugerindo uma ênfase na

organização jurídica e na gestão de documentos oficiais. Além disso, palavras como "portaria,designar,responder"e "atribuição"apontam para aspectos de estrutura organizacional e atribuições funcionais dentro da Defensoria Pública. Esse conjunto de palavras indica um foco no uso de regulamentações para garantir a integridade, eficiência e transparência nos processos administrativos e legais.

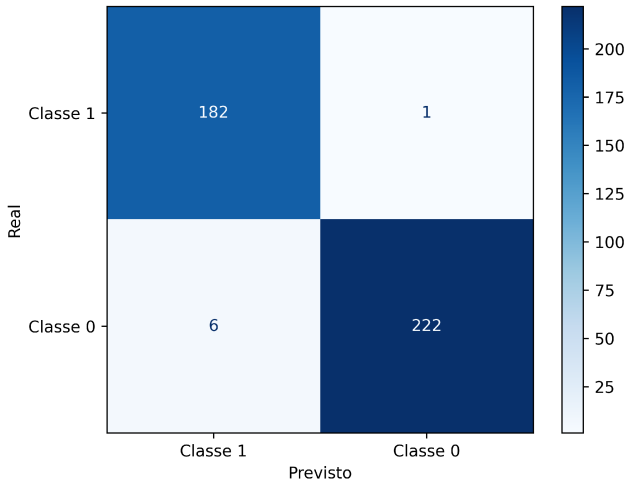
Figura 7 – Nuvem de palavras



Fonte: Elaborada pelo autor.

Com o objetivo de analisar o desempenho dos modelos, foram desenvolvidos três modelos de redes neurais utilizando algoritmos de *deep learning*. Esses modelos demonstraram eficiência na tarefa de classificação textual. O primeiro modelo utiliza a arquitetura de RNN para a classificação de documentos textuais, resultando em uma acurácia de 98,30%, com precisão, *recall* e F1-*Score* médios de 96,81%, 99,45% e 98,11%, respectivamente. A matriz de confusão do modelo Figura 8 mostra que a maioria das classificações corretas ocorreu nas categorias com maior volume de dados de treinamento, evidenciando a importância da quantidade e qualidade dos dados para o desempenho do modelo.

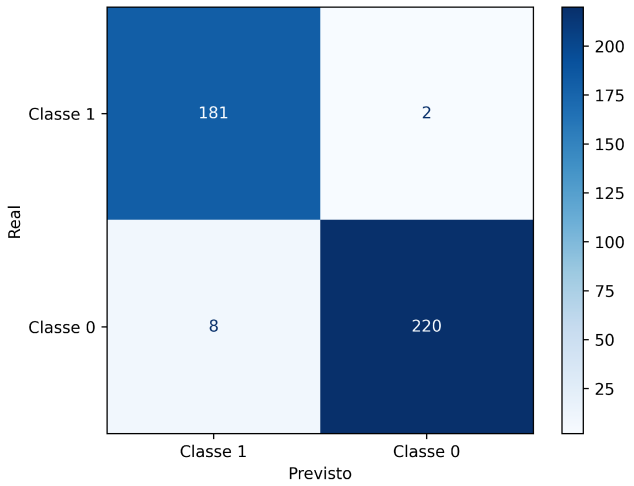
Figura 8 – Matriz de Confusão do Modelo RNN.



Fonte: Elaborada pelo autor.

O segundo modelo emprega a arquitetura de LSTM. Os resultados obtidos com a rede LSTM foram promissores, como demonstrado na matriz de confusão apresentada na Figura 9. As métricas obtidas pelo modelo foram: 97,57% de acurácia, 95,77% de precisão, 98,91% de *recall* e 97,31% de F1-*Score*. Esses resultados indicam que a LSTM é eficaz na captura da semântica nos dados textuais, contribuindo para a classificação.

Figura 9 – Matriz de Confusão do Modelo LSTM.

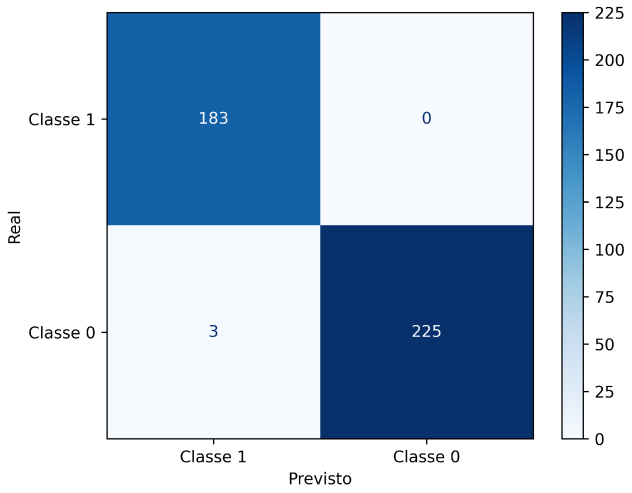


Fonte: Elaborada pelo autor.

O terceiro modelo analisado é o BERT, que demonstrou ser bastante eficiente, alcançando os melhores resultados entre os modelos testados. O BERT obteve uma acurácia de 99,27%, com precisão de 98,39%, *recall* de 100% e F1-*score* de 99,19%. A matriz de confusão do modelo Figura 10 destaca a superioridade do BERT em relação aos modelos LSTM

e RNN, especialmente na capacidade de identificar corretamente todas as instâncias da classe positiva.

Figura 10 – Matriz de Confusão do Modelo BERT.



Fonte: Elaborada pelo autor.

5.3 Avaliação dos Resultados

Com o objetivo de verificar o desempenho da abordagem TF-IDF, os mesmos dados utilizados nos três modelos de aprendizado profundo, foram utilizados com esse modelo e durante os testes foi possível avaliar os resultados dessa técnica tradicional. Os resultados obtidos durante os testes podem ser observados abaixo na Tabela 4. Foram utilizadas as métricas de acurácia, precisão, *recall* e F1-*Score* para avaliar o desempenho do modelo. Os resultados obtidos com este modelo foram satisfatórios. No entanto, devido o grande volume e a complexidade de dados que criados nos dias atuais, é preciso avaliar e escolher qual modelo utilizar. Pois os modelos mais tradicionais, não são tão ajustáveis quanto os modelos mais complexos como o caso do modelos de aprendizado profundo.

Tabela 4 – Métricas do modelo NB com TF-IDF.

Modelo	**Acurácia**	**Precisão**	**Recall**	**F1-Score**
Modelo NB	98,05	100,00	96,55	98,24

Fonte: Elaborada pelo autor.

Na seção de trabalhos relacionados, foram apresentados e discutidos resultados sobre abordagens tradicionais, como é o caso da abordagem TF-IDF aplicada à tarefa de classificação de textos em trabalhos anteriores. A partir dos resultados obtidos, foi possível realizar uma análise comparativa do desempenho em relação ao TF-IDF e os modelos de *deep learning*, evidenciando a eficiência do modelo BERT em relação a abordagem tradicional e aos outros modelos baseados também em arquitetura de rede profunda, como

LSTM e RNN. Embora as diferenças de desempenho entre esses modelos sejam pequenas, a capacidade superior do BERT em lidar com tarefas complexas está bem documentada na literatura.

Apesar de pequena a diferença, o BERT demonstrou vantagem em relação a abordagens tradicionais de aprendizado de máquina, especialmente na tarefa de classificação de textos. Podemos observar essa diferença observando a métrica do F1-*Score* em relação aos outros modelos, pois a base de dados é desbalanceada e nesse caso, o modelo BERT teve uma leve vantagem. Entretanto, verificou-se que o BERT necessita de um conjunto de dados robusto para garantir uma boa generalização, sendo mais suscetível ao *overfitting* quando aplicado a conjuntos de dados menores. Estudos anteriores corroboram essas observações, evidenciando que a integração de técnicas de PLN com *deep learning*, como o BERT, proporciona melhorias substanciais em precisão e eficiência, especialmente na automação de processos em órgãos públicos. Para mitigar o risco de *overfitting*, é essencial adotar técnicas que garantam a generalização adequada do modelo, conforme observado nos dados experimentais.

Conforme pode ser observado na Tabela 5, o BERT apresentou desempenho superior em relação aos modelos LSTM e RNN, especialmente nas métricas de acurácia e *recall*. O BERT atingiu uma acurácia de 99,27%, superando os modelos LSTM (97,57%) e RNN (98,30%), demonstrando uma capacidade de generalização mais robusta. O *recall* de 100% destaca a habilidade do BERT de identificar todas as instâncias relevantes, sem deixar de classificar nenhuma corretamente, o que é crucial em cenários onde a detecção completa é necessária.

Na métrica precisão do BERT, o modelo também foi superior com o percentual (98,39%) em comparação com os modelos LSTM (95,77%) e RNN (96,81%). Já a métrica *recall*, foi %100, o que significa que o modelo não falha em capturar as instâncias relevantes, apesar de gerar alguns falsos positivos. Esse compromisso entre precisão e *recall* é refletido no F1-*Score* de 99,19%, que combina essas duas métricas e posiciona o BERT como o modelo com o desempenho mais equilibrado.

Esses resultados indicam que o BERT é particularmente eficaz em tarefas que exigem alta capacidade de detecção (alto *recall*) sem comprometer significativamente a precisão. Estudos na literatura confirmam que, em aplicações onde a detecção correta de todas as instâncias de uma classe é essencial, como nas áreas médica ou jurídica, o BERT oferece vantagens significativas em relação a modelos mais simples, como LSTM e RNN. Portanto, o desempenho do BERT, conforme evidenciado pelas métricas, é adequado para cenários complexos que exigem um equilíbrio entre acurácia geral e detecção completa das instâncias positivas.

Tabela 5 – Métricas dos modelos.

Métrica	**Rede RNN**	**Rede LSTM**	**Rede BERT**	**Média ± Desvio Padrão**
Acurácia	98,30	97,57	99,27	98,38 ± 0,70
Precisão	96,81	95,77	98,39	96,99 ± 1,08
Recall	99,45	98,91	100,00	99,45 ± 0,45
F1-Score	98,11	97,31	99,19	98,20 ± 0,77

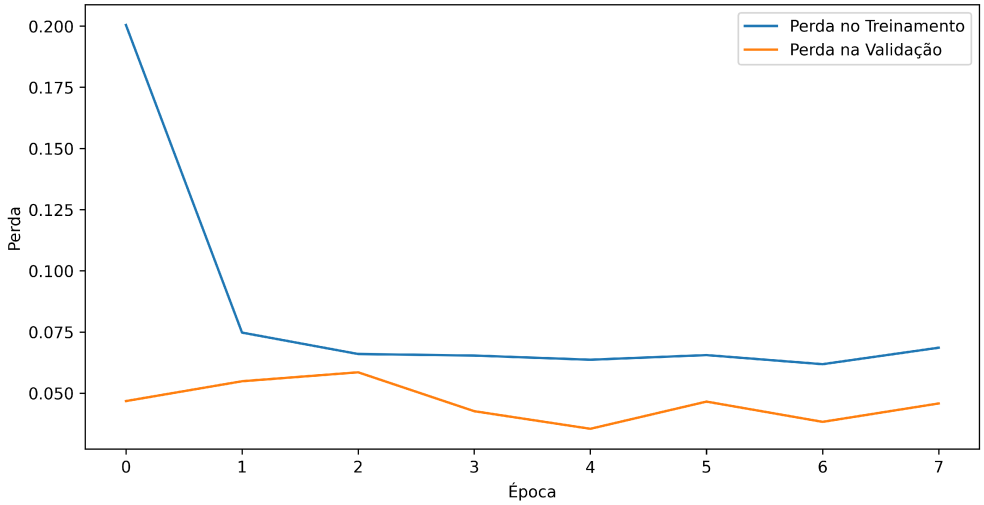
Fonte: Elaborada pelo autor.

5.3.1 Função de perda e acurácia do melhor modelo

No gráfico de perda em relação às épocas Figura 11, a perda de treinamento apresenta uma queda constante ao longo do tempo, indicando que o modelo está aprendendo adequadamente a partir dos dados de treinamento. A perda de validação, no entanto, permanece quase estável nas primeiras épocas e depois começa a subir levemente, o que sugere o início de um *overfitting*. Embora o modelo esteja se ajustando bem aos dados de treinamento, o aumento da perda de validação é um sinal de que ele pode estar começando a memorizar os dados de treinamento, perdendo sua capacidade de generalizar para novos dados.

Isso demonstra o quão poderoso é o modelo BERT baseado na arquitetura Transformer. Durante os testes do modelo, foram implementadas e aplicadas duas técnicas de *Data Augmentation* (DA): a substituição por sinônimos *Random Synonym Replacement* (RSR), que gera variações no texto trocando palavras por sinônimos, mantendo o significado; e a troca aleatória de palavras *Random Word Swapping* (RWS), que modifica a estrutura gramatical do texto sem alterar o significado. Com a aplicação dessas técnicas, foi possível gerar novos dados e observar que o BERT precisa de um conjunto de dados de qualidade e robusto para evitar problemas de *overfitting*.

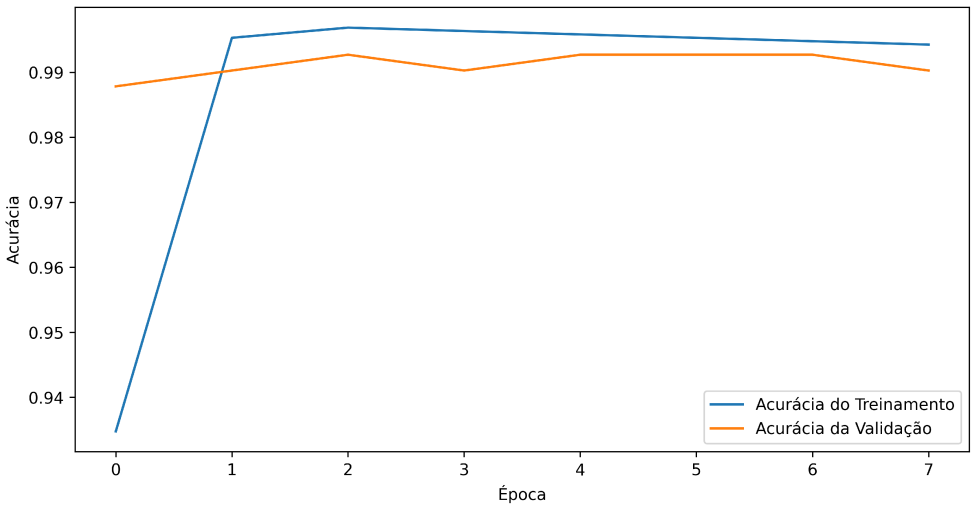
Figura 11 – Função de Perda do Modelo BERT.



Fonte: Elaborada pelo autor.

No gráfico de acurácia em relação às épocas Figura 12, a acurácia de treinamento aumenta rapidamente, se aproximando de 100%, indicando que o modelo consegue classificar quase perfeitamente os dados de treinamento após poucas épocas. No entanto, a acurácia de validação começa em um valor alto e se mantém estável nas primeiras épocas, mas começa a cair ligeiramente, o que também indica sinais de *overfitting*. A queda na acurácia de validação, combinada com o aumento contínuo da acurácia de treinamento, mostra que o modelo está começando a perder sua capacidade de generalização.

Figura 12 – Treinamento e Validação Acurácia do Modelo BERT.



Fonte: Elaborada pelo autor.

Para mitigar o *overfitting* observado nos resultados apresentados na Figura 12, foram aplicadas diversas técnicas. A taxa de *dropout* foi aumentada para 30% para regularizar o modelo; a camada intermediária foi reduzida para 128 neurônios para diminuir sua complexidade; e foi adicionada a *Batch Normalization* para estabilizar o treinamento. Além disso, aplicou-se a regularização L2 (*Ridge*) para evitar o *overfitting* e implementou-se o *Early Stopping* para interromper o treinamento se a perda de validação aumentasse. Por fim, utilizou-se a *Cross-Validation* com 5 *folds* (K-Fold) para garantir a boa generalização do modelo.

Capítulo 6

Conclusão

Ao longo desta dissertação, foi realizada uma pesquisa no campo de classificação de texto utilizando PLN e AP, com foco na classificação de textos jurídicos específicos extraídos de portarias e atos administrativos. Inicialmente, na introdução, destacou-se a crescente importância dessas técnicas e a relevância de explorar novas abordagens para a obtenção automatizada de informações textuais em conjuntos de dados não estruturados presentes em páginas *web*.

A fundamentação teórica proporcionou um embasamento conceitual, abordando desde a coleta de dados e técnicas de PLN até a classificação automática de textos. Essa base teórica permitiu a compreensão das metodologias propostas realizadas e aplicadas em trabalhos relacionados. Com a revisão dos trabalhos relacionados, possível identificar oportunidade de aplicar automação de processos e verificar que houve avanços tecnológicos, sustentando a hipótese do estudo.

A metodologia proposta destacou-se pela combinação estratégica de técnicas de PLN e AP com abordagem supervisionada, concentrando-se especialmente no pré-processamento de dados brutos, na criação de um conjunto de dados rotulado e na classificação eficiente de textos. Os objetivos traçados incluíram o desenvolvimento de um modelo que combina técnicas de PLN e DL aplicadas a textos jurídicos específicos que aborda portarias e atos administrativos, alcançando uma precisão de 100% com o modelo BERT. Esse percentual obtido pelo modelo BERT, de demonstra a robustez e eficiente do modelo na tarefa de classificação de textos associadas à linguagem natural. No entanto, esses modelos são aplicados em problemas com grande volume de dados ou em problemas mais complexos que exige um modelo personalizável ao o problema específico.

A hipótese formulada, baseada na premissa de que a aplicação conjunta dessas técnicas pode revolucionar a coleta, o pré-processamento e a classificação de textos, foram confirmadas à luz da metodologia empregada e dos resultados obtidos. As principais contribuições deste estudo incluem:

1. Desenvolvimento de um modelo eficiente para a classificação de textos jurídicos

utilizando BERT, que alcançou uma acurácia de 99%, demonstrando superioridade em relação a modelos como LSTM (97%) e RNN (98%).

2. Integração de técnicas avançadas de PLN e AP para otimizar as tarefas de pré-processamento e classificação de textos, resultando em modelos com acurácia acima de 95% nos três modelos testados.

3. Criação de um conjunto de dados rotulados, composto por textos do segmento jurídico relacionados à substituições e designações de defensores públicos, isto é, relacionados à portarias e atos administrativos, que será disponibilizado publicamente para a comunidade científica.

4. Demonstração prática dos benefícios da automação de processos na administração pública, evidenciando a aplicabilidade do modelo BERT na otimização de tarefas complexas de classificação de textos jurídicos específicos.

Os resultados obtidos confirmam que o modelo BERT, baseado na arquitetura *Transformer*, não apenas superou outros modelos em termos de acurácia e desempenho geral, mas também reforçou sua aplicabilidade na automação de processos em órgãos públicos. A abordagem proposta oferece melhorias substanciais em termos de precisão e eficiência, destacando-se como uma solução eficaz para a classificação de textos na área jurídica. No entanto, o modelo BERT foi a rede que levou mais tempo para ser treinada.

Em suma, este estudo validou e verificou a possibilidade de automação desde a coleta de dados até a classificação de texto de forma automatizada com aplicação e integração entre PLN e AP. Depois da etapa de pré-processamento dos dados, criou-se uma base rotulada para futuros testes. Espera-se que as contribuições aqui apresentadas sirvam como referência para pesquisas futuras e que o conjunto de dados disponibilizado impulsione novos estudos no campo do processamento de linguagem natural aplicado ao contexto de textos jurídicos com foco em portarias e atos administrativos.

6.1 Principais Contribuições

Os experimentos realizados validaram a hipótese estabelecida, evidenciando as seguintes contribuições significativas:

1. Aplicação e validação de um modelo baseado em BERT que alcançou acurácia de 99,27% na classificação de textos jurídicos específicos, superando abordagem TF-IDF e estabelecendo novos percentuais de desempenho na classificação de textos.

2. Implementação de uma metodologia automática eficaz desde a coleta até a classificação que combina técnicas de PLN e AP, otimizando as etapas de pré-processamento

e classificação, o que resultou em modelos obtendo resultados com acurácia acima de 90% nos três modelos testados (BERT, LSTM e RNN).

3. Disponibilização pública de um conjunto de dados rotulado específico do segmento jurídico com foco em portarias e atos administrativos, promovendo a replicabilidade dos estudos e incentivando novas pesquisas na temática.

4. Demonstração prática da aplicabilidade da metodologia proposta na automação de processos internos em órgãos públicos, evidenciando melhorias substanciais em termos de eficiência e desempenho em relação à análise manual.

6.2 Trabalhos Futuros

Para trabalhos futuros, propõe-se:

1. Explorar a aplicação do modelo desenvolvido em outros domínios jurídicos e ampliar o conjunto de dados para incluir diferentes tipos de documentos legais.

2. Investigar a integração do modelo com sistemas existentes nas instituições públicas para promover à adoção da tecnologia na modernização dos processos administrativos.

3. Avaliar o impacto da automação na rotina dos servidores públicos, considerando aspectos éticos, legais e organizacionais.

4. Incorporar técnicas de balanceamento de classes e aumento de dados, ou *Data Augmentation* do inglês, visando melhorar ainda mais o desempenho dos modelos em cenários com dados limitados.

5. Implementar a integração do melhor e mais viável modelo com os sistemas de informação existentes para promover a indexação automática das informações.

6.3 Contribuições em Produção Bibliográfica

A pesquisa descrita nesta dissertação resultou na submissão de um artigo em um *journal* e a publicação de um resumo expandido.

1. O artigo *Application of Natural Language Processing and Deep Learning in the Task of Legal Text Classification* foi submetido para o *Journal of the Brazilian Computer Society (JBCS)*. Nesse artigo foram reunidas as principais propostas e resultados descritos nesta dissertação.

2. ARAÚJO, David P.; FERNANDES, Henrique C.; PEREIRA, Fabíola S. F. Processamento de linguagem natural e aprendizado de maquina aplicados à extração e indexação automática de informações textuais. In: X FACOM TECHWEEK E XVII WORKSHOP DE TESES E DISSERTAÇÕES EM CIÊNCIA DA COMPUTAÇÃO, 2023, Uberlândia. XVII WTDCC. Uberlândia, 2023. p. 48-49.

Referências

AGARWAL, A.; XU, S.; GRABMAIR, M. Extractive summarization of legal decisions using multi-task learning and maximal marginal relevance. **arXiv preprint arXiv:2210.12437**, 2022. Disponível em: <https://arxiv.org/abs/2210.12437>.

AHMED, H.; TRAORE, I.; SAAD, S. Detecting opinion spams and fake news using text classification. **Security and Privacy**, 2017.

BISONG, E. Google colaboratory. In: _____. **Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners**. Berkeley, CA: Apress, 2019. p. 59–64. ISBN 978-1-4842-4470-8. Disponível em: <https://doi.org/10.1007/978-1-4842-4470-8_7>.

CHAN, C.-P.; YANG, J.-H. Instagram text sentiment analysis combining machine learning and nlp. **ACM Transactions on Asian Low-Resource Language Information Processing**, 2023.

CHERIF, W.; MADANI, A.; KISSI, M. Text categorization based on a new classification by thresholds. **Progress in Artificial Intelligence**, v. 10, p. 205–218, 2021.

CHOLLET, F. **Deep Learning with Python**. [S.l.]: Manning Publications Co, 2021. <https://www.manning.com/books/deep-learning-with-python>.

DENG, L.; LIU, Y. **Deep Learning in Natural Language Processing**. Seattle: Springer, 2018.

DEVLIN, J. et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In: **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**. [S.l.]: Association for Computational Linguistics, 2019. p. 4171–4186.

FAROKHIAN, M.; RAFE, V.; VEISI, H. Fake news detection using dual bert deep neural networks. **Multimedia Tools and Applications**, 2023.

FERREIRA, H. H. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação), **Processamento de linguagem natural e classificação de textos em sistemas modulares**. Brasília: [s.n.], 2019.

FIGUEIREDO, F. S. **Construção de evidências para classificação automática de textos**. Dissertação (Dissertação de Mestrado em Ciência da Computação) — Universidade Federal de Minas Gerais, Belo Horizonte, 2008.

FLAYEH, A. K.; HAMODI, Y. I.; ZAKI, N. D. Text analysis based on natural language processing (nlp). In: **Proceedings of the 2nd International Conference on Advances in Engineering Sciences and Technology (AEST-2022)**. University of Babylon: IEEE, 2022. p. 774–780.

GONçALVES, E. C. Introdução à classificação multirrótulo. In: **Anais da V Escola Regional de Sistemas de Informação do Rio de Janeiro**. [S.l.]: Sociedade Brasileira de Computação, 2018.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GRUS, J. **Data Science do Zero: primeiras regras com o Python**. [S.l.]: Alta Books, 2016. <https://altabooks.com.br/produto/data-science-do-zero/>.

JIANG, T. et al. Health education based on natural language processing (nlp) for infectious disease outbreak. In: **Proceedings of the 2nd International Conference on Artificial Intelligence and Education (ICAIE)**. Dali: IEEE, 2021. p. 667–670.

JORDAN, M. I.; MITCHELL, T. M. Machine learning: Trends, perspectives, and prospects. **Science**, v. 349, n. 6245, p. 255–260, 2015.

JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models**. [S.l.]: Online manuscript released August 20, 2024, 2024. <https://web.stanford.edu/~jurafsky/slp3>.

LAVANYA, P. M.; SASIKALA, E. Deep learning techniques on text classification using natural language processing (nlp) in social healthcare network: A comprehensive survey. In: **Proceedings of the 3rd International Conference on Signal Processing and Communication (ICSPC)**. Coimbatore: IEEE, 2021. p. 318–322.

MCKINNEY, W. **Python for Data Analysis Data Wrangling with pandas, NumPy, and Jupyter**. [S.l.]: O'Reilly Media, Inc, 2022. <https://www.oreilly.com/library/view/python-for-data/9781098104023/>.

MIAO, F. et al. Chinese news text classification based on machine learning algorithm. In: **Proceedings of the 2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)**. [S.l.]: IEEE, 2018. p. 48–51.

MOHANA, P. et al. Automation using artificial intelligence based natural language processing. In: **Proceedings of the 6th International Conference on Computing Methodologies and Communication (ICCMC)**. Salem: IEEE, 2022.

MULAHUWAISH, A. et al. Efficient classification model of web news documents using machine learning algorithms for accurate information. **Computers & Security**, v. 98, p. 102006, 2020.

OLIVEIRA, F. et al. Avanços em redes neurais lstm e aplicações em processamento de linguagem natural. **Journal of Computational Intelligence**, v. 15, n. 1, p. 85–104, 2022.

PIVETTA, S. P.; MERGEN, S. L. S.; KEPLER, F. N. Uso de aprendizado de máquina para a classificação de documentos do exército brasileiro. In: **Anais do 9º Simpósio Brasileiro de Sistemas de Informação (SBSI)**. Porto Alegre: Sociedade Brasileira de Computação, 2013. p. 768–779.

PLAAT, A. **Deep Reinforcement Learning: Fundamentals, Research and Applications**. Cham: Springer, 2022.

RANGEL, M. et al. Uso de aprendizado de máquina para categorização automática de conjuntos de dados de portais de dados abertos. In: **Anais do 8º Workshop de Computação Aplicada em Governo Eletrônico (WCGE)**. Porto Alegre: Sociedade Brasileira de Computação, 2020. p. 120–131. ISSN 2763-8723.

RAO, C. S. et al. Effective fake news classification based on lightweight rnn with nlp. **Annals of Data Science**, 2023.

_____. Effective fake news classification based on lightweight rnn with nlp. **Annals of Data Science**, 2024.

SEBASTIANI, F. Machine learning in automated text categorization. **ACM Computing Surveys**, v. 34, n. 1, p. 1–47, 2002.

SETT, S.; SINGH, A. V. Applying natural language processing in healthcare using data science. In: **Proceedings of the 11th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)**. Noida: IEEE, 2024.

SHARMA, H.; DEVI, B. Advancements in natural language processing: Techniques and applications. In: **Proceedings of the International Conference on Advanced Computing & Communication Technologies**. Mohali: IEEE, 2023.

SOUZA, P.; SILVA, J. Redes neurais recorrentes no processamento de sequências: Revisão e avanços recentes. **Revista Brasileira de Computação**, v. 10, n. 2, p. 45–62, 2021.

STEVENS, E.; ANTIGA, L.; VIEHMANN, T. **Deep Learning with PyTorch**. [S.l.]: Manning Publications Co, 2020. <https://www.manning.com/books/deep-learning-with-pytorch>.

SUTTON, R. S.; BARTO, A. G. **Reinforcement Learning: An Introduction**. 2. ed. [S.l.]: MIT Press, 2018.

VASILIEV, Y. **Natural language processing with Python and spaCy: A practical introduction**. [S.l.]: No Starch Press, 2020. <https://www.oreilly.com/library/view/natural-language-processing/9781098122652/>.

WAN, M. et al. Tnt-llm: Text mining at scale with large language models. In: **Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining**. Barcelona: ACM, 2024. p. 1–12.

WANG, B. Disconnected recurrent neural networks for text categorization. In: **Proceedings of the Annual Meeting of the Association for Computational Linguistics**. Melbourne: ACL, 2018. p. 2311–2320.

WANG, X.; KIM, H.-C. Text categorization with improved deep learning methods. **Journal of Information and Communication Convergence Engineering**, v. 16, n. 2, p. 106–112, 2018.

WEHRMANN, J.; KOLLING, C.; BARROS, R. C. Fast and efficient text classification with class-based embeddings. In: **2019 International Joint Conference on Neural Networks (IJCNN)**. [S.l.]: IEEE, 2019. p. 1–8.

WU, H.; LIU, Y.; WANG, J. Review of text classification methods on deep learning. **Computers, Materials & Continua**, v. 63, n. 3, 2020.

ZHANG, B. News text classification algorithm based on machine learning technology. In: **Proceedings of the 2022 International Conference on Education, Network and Information Technology (ICENIT)**. [S.l.]: IEEE, 2022. p. 182–186.

ZHANG, J.; NIU, Y.; NIE, H. Web document classification based on fuzzy k-nn algorithm. In: **Proceedings of the International Conference on Computational Intelligence and Security (CIS)**. [S.l.: s.n.], 2009. v. 1, p. 193–196.

Apêndices

APÊNDICE A

Códigos utilizados

Nesta seção, são apresentados os códigos implementados para realizar a coleta de dados, a conversão de arquivos em formato PDF para texto, a rotulação preliminar, a limpeza e a criação do dataset inicial.

A.1 Coleta dos dados em PDF

A Figura 13 apresenta o código completo implementado e utilizado para realizar a coleta de dados usando as técnicas apresentadas e detalhadas na seção de Materiais e Métodos.

A.2 Conversão dos arquivos PDF para texto

A Figura 14 apresenta o código completo implementado e utilizado para realizar a conversão dos dados extraídos dos arquivos PDF para o arquivo de texto e posteriormente criar o *dataset* usando as bibliotecas apresentadas com mais detalhes em outras seções deste trabalho.

Figura 13 – Código para realizar a coleta de dados

```python
import requests
from bs4 import BeautifulSoup
import os

class PDFCrawler:
    def __init__(self, start_page=1, end_page=1, output_folder='downloads'):
        self.base_url = 'http://diariooficial.defensoria.to.def.br/'
        self.url_template = 'http://diariooficial.defensoria.to.def.br/editions?page={}'
        self.start_page = start_page
        self.end_page = end_page
        self.output_folder = output_folder
        self.headers = {'User-Agent': 'Mozilla/5.0'}
        os.makedirs(self.output_folder, exist_ok=True)

    def crawl(self):
        for page_number in range(self.start_page, self.end_page + 1):
            print(f'Processando página {page_number}')
            page_url = self.url_template.format(page_number)
            response = requests.get(page_url, headers=self.headers)
            if response.status_code != 200:
                print(f'Erro ao acessar a página {page_number}')
                continue
            soup = BeautifulSoup(response.content, 'html.parser')
            pdf_links = self.extract_pdf_links(soup)
            if pdf_links:
                self.download_pdfs(pdf_links)
            else:
                print(f'Nenhum link para arquivo PDF encontrado na página {page_number}.')

    def extract_pdf_links(self, soup):
        pdf_links = []
        links = soup.find_all('a', href=True, attrs={"target": "blank", "title": "Download"})
        for link in links:
            href = link.get('href')
            if href and href.endswith('.pdf'):
                pdf_links.append(self.base_url + href)
        return pdf_links

    def download_pdfs(self, pdf_links):
        for link in pdf_links:
            pdf_filename = link.split('/')[-1]
            pdf_path = os.path.join(self.output_folder, pdf_filename)
            if os.path.exists(pdf_path):
                print(f'O arquivo {pdf_filename} já existe. Pulando o download.')
                continue
            try:
                pdf_response = requests.get(link)
                with open(pdf_path, 'wb') as pdf_file:
                    pdf_file.write(pdf_response.content)
                print(f"Arquivo {pdf_filename} baixado e salvo em: {pdf_path}")
            except Exception as e:
                print(f'Erro ao baixar o arquivo {pdf_filename}: {e}')

if __name__ == '__main__':
    # Solicita ao usuário o número da página inicial e final
    start_page = int(input('Digite o número da página inicial: '))
    end_page = int(input('Digite o número da página final: '))

    # Cria uma instância da classe PDFCrawler com os parâmetros fornecidos
    crawler = PDFCrawler(start_page=start_page, end_page=end_page)
    crawler.crawl()
```

Fonte: Elaborada pelo autor.

Figura 14 – Código para realizar a conversão de arquivos PDF para texto

```python
# !pip install PyPDF2

import os
import PyPDF2
import pandas as pd

class PDFTextExtractor:
    def __init__(self, input_folder='downloads', output_csv='output_file.csv'):
        self.input_folder = input_folder
        self.output_csv = output_csv
        self.keywords = [
            "DESIGNAR",
            "REVOGAR",
            "PRORROGAR",
            "ALTERAR",
            "SUSPENDER"
        ]
        self.data = []

    def extract_text_from_pdf(self, pdf_path):
        text = ""
        try:
            with open(pdf_path, "rb") as f:
                pdf_reader = PyPDF2.PdfReader(f)
                for page_num in range(len(pdf_reader.pages)):
                    page = pdf_reader.pages[page_num]
                    # Mantém a formatação dos parágrafos
                    text += page.extract_text() + "\n\n"
        except Exception as e:
            print(f"Erro ao converter {pdf_path}: {str(e)}")
        return text

    def label_text(self, paragraph):
        for keyword in self.keywords:
            if keyword in paragraph:
                return 1
        return 0

    def process_pdfs(self):
        # Percorre todos os arquivos PDF na pasta de entrada
        for filename in os.listdir(self.input_folder):
            if filename.endswith('.pdf'):
                input_pdf_path = os.path.join(self.input_folder, filename)
                text = self.extract_text_from_pdf(input_pdf_path)

                # Divide o texto em parágrafos
                paragraphs = text.split("\n\n")
                for paragraph in paragraphs:
                    # Remove espaços em branco no início e no fim
                    paragraph = paragraph.strip()
                    if paragraph:
                        label = self.label_text(paragraph)
                        self.data.append({'texto': paragraph, 'rotulo': label})

    def save_to_csv(self):
        # Cria um DataFrame do pandas com os dados coletados
        df = pd.DataFrame(self.data)
        # Salva o DataFrame em um arquivo CSV
        df.to_csv(self.output_csv, index=False, encoding='utf-8')
        print(f"Arquivo CSV salvo em: {self.output_csv}")

    def run(self):
        self.process_pdfs()
        self.save_to_csv()
        print("Processamento concluído.")

if __name__ == "__main__":
    # Define a pasta de entrada e o nome do arquivo CSV de saída
    input_folder = 'downloads'
    output_csv = 'output_file.csv'

    extractor = PDFTextExtractor(input_folder=input_folder, output_csv=output_csv)
    extractor.run()
```

Fonte: Elaborada pelo autor.