UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel Teles

Uso de Redes Neurais Profundas para Detecção de Fake News

Uberlândia, Brasil

2023

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel Teles

Uso de Redes Neurais Profundas para Detecção de Fake News

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informação.

Orientador: Fernanda Maria da Cunha Santos

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2023

Gabriel Teles

Uso de Redes Neurais Profundas para Detecção de Fake News

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informação.

Trabalho aprovado. Uberlândia, Brasil, 04 de dezembro de 2023:

Fernanda Maria da Cunha Santos
Orientador

Daniel Stefany Duarte Caetano

Paulo Henrique Ribeiro Gabriel

Uberlândia, Brasil
2023

Dedico a

Deus, fonte de sabedoria e força, dedico este trabalho como expressão da gratidão pela jornada acadêmica e pelas bênçãos recebidas ao longo dela.

Aos meus pais, Geraldo Luciano Teles e Andrea Neris Pereira Teles, agradeço por serem os pilares da minha vida. Seu amor, apoio e sacrifícios moldaram quem sou e são a base deste sucesso.

Aos meus irmãos, Miguel, Franciele e Martinha, agradeço pela cumplicidade, compreensão e alegrias compartilhadas ao longo desta trajetória.

Aos meus queridos amigos, que foram mais do que companheiros de estudo, mas também fontes de inspiração e apoio nos momentos desafiadores, a dedicação deste trabalho é um reflexo do valor que vocês têm em minha vida.

Que este trabalho seja não apenas uma conquista individual, mas uma celebração compartilhada com aqueles que têm um lugar especial em meu coração.

Agradecimentos

Gostaria de dedicar um agradecimento especial à minha orientadora, Professora Fernanda Maria da Cunha Santos, cuja contribuição foi essencial para a realização deste trabalho.

Professora Fernanda, sua orientação diligente e comprometida desempenhou um papel crucial na definição da direção desta pesquisa. Sua experiência e conhecimento foram uma fonte valiosa de inspiração, orientando-me através dos desafios acadêmicos com paciência e dedicação.

Agradeço não apenas pela sua competência profissional, mas também pela sensibilidade em compreender minhas dúvidas e incertezas, proporcionando um ambiente propício para o crescimento acadêmico e pessoal.

Este trabalho reflete não apenas minhas próprias realizações, mas também a orientação excepcional que recebi de uma mentora dedicada. Agradeço por sua disponibilidade, orientação crítica e constante encorajamento.

Obrigado, Professora Fernanda, por ser uma presença inspiradora ao longo desta jornada acadêmica .

Resumo

Este documento aborda a detecção de notícias falsas (*fake news*) por meio da aplicação de redes neurais de *aprendizado profundo*. O objetivo principal é desenvolver um sistema eficaz de detecção automática que possa discernir entre notícias reais e fabricadas. Uma base de dados pública contendo textos digitais verdadeiros e falsos, foi utilizada como fonte de informação, e os modelos proposto para o sistema computacional foram formados por técnicas de processamento de linguagem natural, principalmente, o *Word2Vec* e diferentes redes neurais *LSTM (Long Short-Term Memory)* como a responsável pela classificação dos textos. Os resultados revelam uma notável melhoria na precisão da detecção em comparação à outros métodos. As conclusões destacam a importância do uso de técnicas avançadas de processamento de linguagem natural e aprendizado profundo na mitigação do impacto das notícias falsas.

Palavras-chave: *Fake news*, *PLN*, *Deep learning*, *Word2Vec*, *LSTM*.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AM	Aprendizado de Máquinas
CBOW	*Continuous Bag of Words*
IA	Inteligência Artificial
LSTM	*Long Short Term Memory*
NLP	*Natural Language Processing*
PLN	Processamento de Linguagem Natural
RNA	Rede Neural Artificial
RNN	*Recurrent Neural Network*

Sumário

1 Introdução

A disseminação de *fake news*, ou notícias falsas, tornou-se um problema significativo nos últimos anos, com o potencial de influenciar eleições (RODRIGUES; BONONE; MIELLI, 2020), prejudicar a reputação de indivíduos e organizações, e até colocar vidas de pessoas em risco (TEIXEIRA; SANTOS et al., 2020). A rápida propagação de informações através da internet e das redes sociais facilitou ainda mais que as notícias falsas se espalhassem rapidamente, muitas vezes antes que a verdade possa ser verificada. É por isso que a detecção de notícias falsas é um desafio tão importante.

Atualmente, estão sendo implementadas diversas medidas por governos, empresas de tecnologia e organizações da sociedade civil com o objetivo de identificar e combater a disseminação de notícias falsas. Duas estratégias amplamente utilizadas são a verificação de fatos por organizações especializadas e o uso de algoritmos de detecção automática. A verificação de fatos é realizada por organizações especializadas que analisam e investigam informações antes de divulgá-las ao público, visando fornecer uma avaliação precisa da veracidade das notícias. Já os algoritmos de detecção automática são desenvolvidos para identificar padrões e características de desinformação, permitindo a identificação de conteúdos suspeitos ou falsos. Essas medidas combinadas têm se mostrado eficazes na luta contra as notícias falsas e na promoção de uma informação mais confiável.

Felizmente, a evolução da tecnologia tornou possível usar técnicas de *Natural Language Processing* (NLP), ou *Processamento de Linguagem Natural (PLN)*, que é "um ramo da inteligência artificial (IA) que permite aos computadores compreender, gerar e manipular a linguagem humana"(FORMIGA, 2022), com o *Aprendizado Profundo*, do inglês *Deep Learning* que é (IBM Brasil, 2023), "uma subárea de *machine learning*, que está englobado no contexto da IA e tem como objetivo fornecer informações através da análise de diferentes tipos de dados", para identificar e combater as *notícias falsas* (Oracle, 2023). Os modelos de *PLN* com *Deep Learning* são capazes de aprender a partir de grandes quantidades de dados e identificar padrões complexos em textos que podem indicar a presença de informações falsas.

Esses modelos são capazes de analisar o contexto e as informações contidas em uma notícia, bem como identificar o uso de linguagem enganosa ou sensacionalista. O uso de *Deep Learning* nesses modelos de *PLN* torna-os ainda mais eficientes na identificação de *notícias falsas* , já que eles podem se adaptar e melhorar continuamente com o tempo.

A detecção de *notícias falsas* é uma questão importante para garantir que as pessoas recebam informações precisas e confiáveis. Com o uso de modelos de *PLN* com *Deep Learning*, podemos esperar uma melhoria significativa na detecção de *notícias falsas*

e um aumento na confiabilidade das informações que circulam pela internet e pelas redes sociais.

1.1 Justificativa

O crescimento expressivo de notícias falsas nas redes sociais torna difícil a sua identificação, impossibilitando que organizações, pessoas e até governos respondam rapidamente a tais difamações e reduzam o impacto dessas notícias. Dessa forma, a criação de um sistema computacional que identifique notícias falsas é uma proposta para reduzir a propagação dessas notícias nos meios digitais.

1.2 Objetivo

Desenvolver modelos computacionais para a classificação de notícias falsas em português do Brasil. Esses modelos utilizam técnicas de *PLN* aliadas às Redes Neurais Artificiais (RNAs) da arquitetura *deep learning*.

1.2.1 Objetivo Específico

Fazer um levantamento e análise de diferentes bases de dados disponíveis em língua portuguesa para escolher a base de dados adequada para o problema proposto. Serão considerados critérios como a relevância dos dados, o tamanho da base, a diversidade dos textos e a confiabilidade das fontes. Com base nessa avaliação, será selecionada a base de dados mais adequada para o desenvolvimento do sistema proposto, garantindo assim a qualidade e representatividade das informações utilizadas. A nomenclatura corpus é usada como sinônimo para base de dados.

Analisar e comparar as técnicas de *PLN*. Neste objetivo específico, serão exploradas diferentes técnicas de *PLN* aplicadas ao corpus. A análise e comparação dessas técnicas permitirão identificar as abordagens mais eficientes e adequadas para a resolução do problema proposto, considerando as características da língua portuguesa.

Identificar a melhor arquitetura da rede neural *deep learning*. Será realizado um estudo aprofundado sobre as melhores arquiteturas aplicadas ao processamento de texto. Diferentes arquiteturas de *RNAs*, como as *redes neurais recorrentes (Recurrent Neural Network RNNs)* do tipo *Long Short Term Memory (LSTM)*, serão analisadas e comparadas em termos de desempenho, velocidade de processamento e capacidade de generalização. O objetivo é identificar a arquitetura mais adequada e eficiente para a tarefa proposta, considerando as peculiaridades da língua portuguesa e dos dados disponíveis.

Definir as medidas de avaliação para concluir o sistema proposto. Serão estabelecidas as medidas de avaliação necessárias para aferir o desempenho e a eficácia do sistema proposto. As métricas de avaliação tradicionais são *precision (precisão), recall(revocação), F1-score e accuracy (acurácia)*. Além disso, serão realizados experimentos e testes com conjuntos de validação e teste para verificar o comportamento do sistema em diferentes cenários. Ao final dessa etapa, será possível concluir a efetividade do sistema proposto, bem como identificar possíveis melhorias e ajustes a serem realizados.

1.3 Metodologia

A metodologia para desenvolver um modelo de processamento de linguagem natural com aprendizado profundo para identificar notícias falsas pode ser dividida em algumas etapas principais:

- Coleta de dados: O primeiro passo é coletar um conjunto de dados de notícias, que deve incluir tanto notícias verdadeiras quanto falsas. Esses dados podem ser obtidos a partir de fontes confiáveis, como sites de notícias estabelecidos e bancos de dados públicos. No contexto específico do idioma português, um exemplo de corpus que pode ser utilizado é o Fake.Br disponível em Monteiro et al. (2018).

- Preparação dos dados: O conjunto de dados coletados precisa ser pré-processado para que possa ser usado em um modelo de redes neurais. Isso inclui a limpeza dos dados, como remoção de pontuações e caracteres especiais, e a transformação dos dados em um formato apropriado para o modelo, como vetores de palavras. Para isso foi utilizado técnicas consolidadas na literaturas que *PLN*, como *Word2vec*.

- Criação do modelo: A próxima etapa é criar um modelo de processamento de linguagem natural com *Deep learning*. Existem várias arquiteturas de redes neurais que podem ser utilizadas, como *redes neurais convolucionais (Convolutional Neural Network CNN)* que são comumente usados em processamento de imagem, mas também podem ser aplicados a texto. Eles usam filtros convolucionais para extrair características importantes do texto, como palavras-chave e frases, e podem ser eficazes na identificação de padrões em notícias falsas. As RNA recorrentes (*Recurrent Neural Network* - RNN) têm memória de curto prazo que lhes permite levar em consideração o contexto anterior ao analisar cada palavra em uma frase ou parágrafo. A arquitetura escolhida foi a do tipo *LSTM*, que são RNA recorrentes com memória de longo prazo.

- Treinamento do modelo: O modelo criado precisa ser treinado com o conjunto de dados preparado. Durante o treinamento, o modelo aprende a identificar padrões e características específicas que distinguem as notícias verdadeiras das falsas.

- Avaliação do modelo: Depois de treinado, o modelo precisa ser avaliado para determinar sua precisão na identificação de notícias falsas. Isso pode ser feito usando um conjunto de dados de teste, que não foi utilizado durante o treinamento.
- Aperfeiçoamento do modelo: Com base nos resultados da avaliação, o modelo pode ser aperfeiçoado, refinando a arquitetura do modelo e ajustando os hiperparâmetros.
- Implantação do modelo: Por fim, o modelo aperfeiçoado pode ser implantado para identificar notícias falsas em tempo real, analisando notícias à medida que são publicadas e marcando as notícias falsas para revisão e possível remoção.

1.4 Organização

O presente trabalho está organizado da seguinte forma: o Capítulo 2 descreve conceitos essenciais para o bom entendimento do trabalho, assim como, as principais referências bibliográficas; o Capítulo 3 contém a descrição das etapas do modelo computacional proposto e as características do corpus; o Capítulo 4 mostra os resultados alcançados com os modelos, bem como uma análise quantitativa e comparativa dos mesmos; e por fim, o Capítulo 5 detalha a conclusão do trabalho e as considerações finais.

2 Revisão Bibliográfica

2.1 Notícias Falsas

Segundo Alves e Maciel (2020), notícias falsas são informações falsas ou distorcidas que se espalham entre a população como se fossem verdadeiras, geralmente por meio de mídias sociais ou aplicativos de mensagens.

O fenômeno das notícias falsas tem se tornado uma preocupação global devido ao seu potencial impacto na sociedade. A disseminação dessas informações falsas ocorre de maneira rápida e ampla, aproveitando-se da facilidade de compartilhamento proporcionada pelas mídias sociais e aplicativos de mensagens. Com a viralização das notícias falsas, é comum que muitas pessoas sejam expostas a conteúdos distorcidos, sem saber discernir entre o que é verdadeiro e o que é falso (ALVES; MACIEL, 2020). Esse cenário gera uma atmosfera de desinformação que pode afetar negativamente a confiança na mídia e comprometer a qualidade do debate público.

Combater a disseminação de notícias falsas é um desafio complexo e multidimensional. Requer esforços conjuntos de diferentes atores sociais, incluindo governos, empresas de tecnologia, organizações da sociedade civil e indivíduos. Além de investimentos em tecnologias avançadas para detecção automática de notícias falsas, é fundamental promover a educação e a conscientização da população sobre os riscos e consequências das notícias falsas. Ao capacitar as pessoas a serem mais críticas em relação às informações que consomem e ao incentivar práticas de checagem de fatos, é possível fortalecer a resiliência da sociedade diante das notícias falsas e promover uma cultura de informação confiável e responsável.

2.2 Processamento de Linguagem Natural

PLN é o estudo dos princípios e métodos que permitem a comunicação entre humanos e computadores por meio de linguagem natural (BIRD; KLEIN; LOPER, 2009). *PLN* envolve diversas áreas do conhecimento, como linguística, ciência da computação, IA, matemática e estatística. *PLN* tem diversas aplicações práticas, como tradução automática, sumarização de textos, análise de sentimentos, extração de informação, entre outras (BIRD; KLEIN; LOPER, 2009).

No amplo universo do *PLN* , destaca-se a escolha do algoritmo *Word2Vec* para esta análise. Dentro do vasto conjunto de técnicas e abordagens disponíveis no *PLN*, o *Word2Vec* foi selecionado devido à sua eficácia e capacidade única de representar palavras

em vetores densos, capturando nuances semânticas e sintáticas.

2.2.1 Word2Vec

O *Word2Vec* é um algoritmo que emprega *RNA* (*Self-Supervised*) com o propósito de representar cada palavra por meio de *word embeddings*, que são pesos próprios dele. O que o distingue dos primeiros algoritmos de *embeddings* é a abordagem com a qual ele enfrenta o problema. Para o *Word2Vec*, o contexto é fundamental na definição do significado de uma palavra (EDRONE, 2023). Na Figura 1, é possível observar como ela adquire significados distintos, mesmo utilizando a palavra "manga"nas duas primeiras frases. Na primeira, refere-se à "manga da roupa"(OGUNDEPO, 2021), enquanto na segunda à "manga fruta". Nas duas últimas frases, mesmo com palavras diferentes, os contextos conferem o mesmo significado.

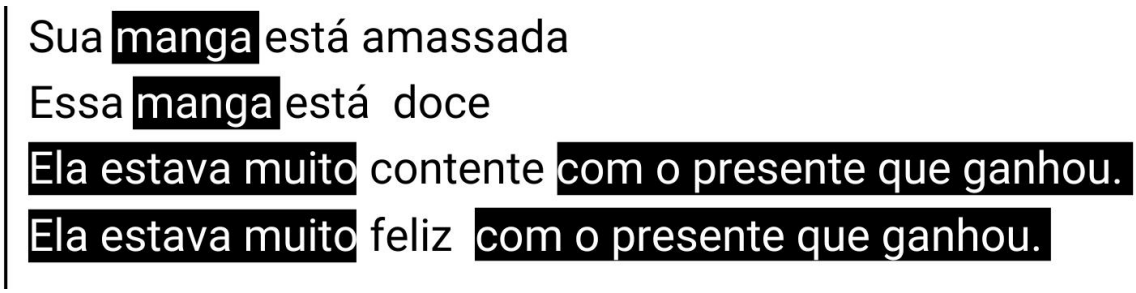

Figura 1 – Exemplo de homônimos e sinônimos ilustrando a importância do contexto para o *Word2Vec*.

Essa capacidade do *Word2Vec* de considerar o contexto ao determinar o significado das palavras é crucial para a precisão e eficácia das representações geradas.

A ideia fundamental por trás da criação desses *word embeddings* reside na capacidade de calcular a probabilidade de uma palavra estar no mesmo contexto de outra. O *Word2Vec*, por exemplo, emprega um modelo que aprende a representação vetorial das palavras com base em probabilidades condicionais.

Para alcançar esse objetivo, o *Word2Vec* utiliza duas arquiteturas principais: *CBOW* (*Continuous Bag of Words*) e *Skip-gram*. No modelo *CBOW*, o propósito é prever a palavra alvo com base no contexto ao seu redor, enquanto no *Skip-gram*, o modelo tenta antecipar o contexto (ou seja, as palavras circundantes) a partir de uma palavra de entrada (EDRONE, 2023).

Essa abordagem possibilita que palavras com contextos semelhantes tenham representações vetoriais mais próximas no espaço de *embedding*. Dessa forma, são capturadas relações semânticas e sintáticas entre as palavras. Tais representações densas e contínuas facilitam operações matemáticas e análises semânticas, sendo aplicáveis a diversas tarefas em processamento de linguagem natural e aprendizado de máquina.

Existem dois conceitos adicionais importantes para compreender o *Word2Vec*, que merecem ser mencionados: a janela de contexto e o vetor *one-hot*. A janela de contexto refere-se a um número natural que representa quantas palavras antes e depois de uma palavra central serão utilizadas como contexto para o *Word2Vec*. No caso do Skip-gram, esse número indica quantas palavras serão previstas, enquanto no CBOW, representa quantas palavras são utilizadas para prever uma palavra central.

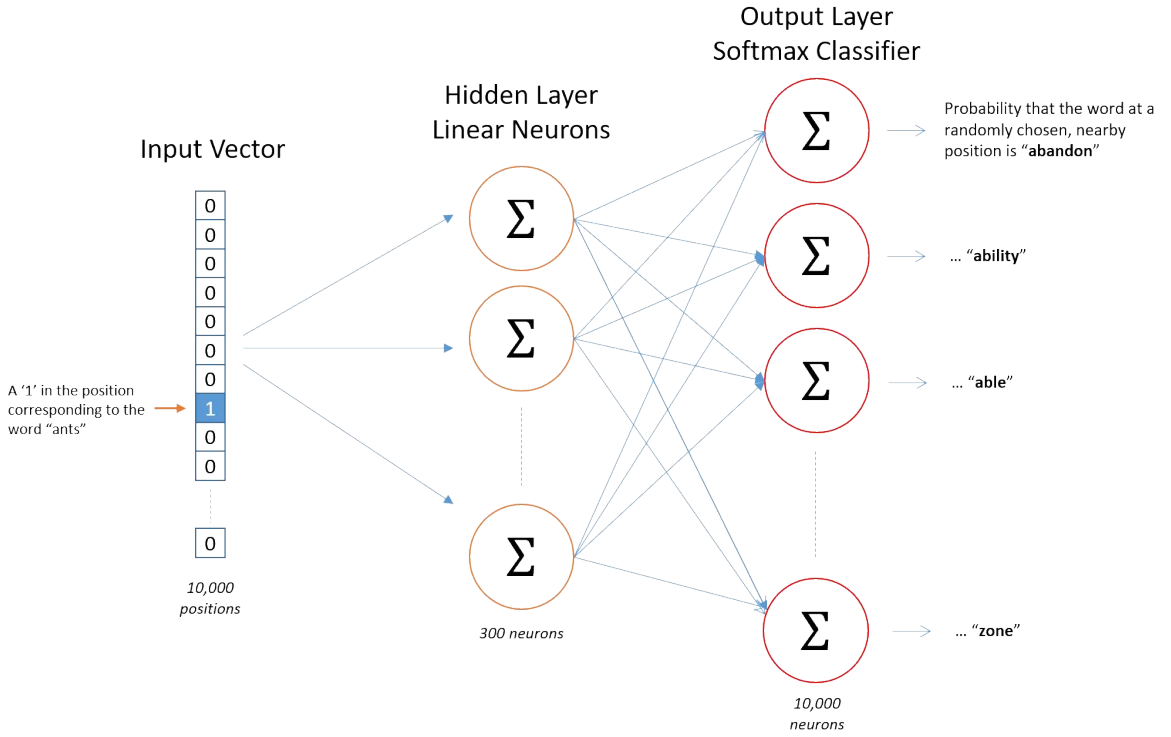

Figura 2 – Arquitetura do *Word2Vec*

Fonte: McCormick (2016)

Além disso, no contexto do *Word2Vec*, todas as palavras do corpus são representadas pela identidade do número de palavras por meio de vetores *one-hot*. Cada vetor *one-hot* é uma coluna dessa matriz, e cada posição no vetor representa uma palavra única (MCCORMICK, 2016).

Esses conceitos, como a janela de contexto e os vetores *one-hot*, desempenham papéis cruciais na definição do escopo e na operação eficiente do *Word2Vec*, permitindo a captura eficaz de relações semânticas e sintáticas entre as palavras no corpus.

A Figura 2 ilustra a arquitetura do *Word2Vec*, composta por três camadas, todas densas. A primeira camada consiste em uma entrada representada por um vetor *one-hot*. Em seguida, há uma camada oculta com n neurônios, resultando em um *word embedding* de n dimensões. Por fim, temos a camada de saída, que é uma camada *softmax*, gerando a probabilidade para cada palavra do dicionário.

Essa estrutura de *RNA* permite a transformação eficiente de representações *one-*

hot em *word embeddings*, onde as relações semânticas e sintáticas são capturadas em um espaço vetorial de dimensão n. A camada *softmax* no final da arquitetura é crucial para gerar as probabilidades associadas a cada palavra no vocabulário, facilitando a aplicação do *Word2Vec* em tarefas diversas de processamento de linguagem natural.

2.3 Deep Learning

De acordo com informações fornecidas pelo site da IBM Brasil (2023), o *Deep Learning* é uma subárea do Aprendizado de Máquina (AM), inserida no contexto mais amplo da IA, com o propósito de fornecer informações por meio da análise de diversos tipos de dados (CHAGAS, 2019). Os sistemas de *Deep Learning* operam de maneira análoga ao cérebro humano, envolvendo a troca e processamento de informações entre neurônios para formar uma *RNA* (IBM Brasil, 2023). Nesse processo, as informações são processadas em camadas, com a primeira camada destinada à entrada de dados para análise e a última exibindo os resultados. Entre essas camadas, encontram-se as camadas escondidas, cujo número varia em cada operação (CHAGAS, 2019).

Ao receber uma informação de entrada, os algoritmos de *Deep Learning* conseguem distinguir detalhes, os quais são submetidos a neurônios que realizam análises por meio de cálculos matemáticos (CHAGAS, 2019). Cada detalhe analisado recebe um peso durante esses cálculos, estabelecendo uma relação com o resultado final esperado. Após os cálculos, o resultado passa por uma função de ativação, que seleciona neurônios a serem ativados, introduzindo não-linearidade ao processo. Esse ciclo de análises continua até alcançar a última camada de resultados.

O funcionamento eficaz do *Deep Learning* demanda uma quantidade substancial de dados e alta capacidade computacional de CPU (CHAGAS, 2019). Essa tecnologia desempenha um papel crucial no desenvolvimento de tecnologias prominentes, como assistentes virtuais, reconhecimento facial, análise de sentimentos e tradução automática, entre outras.

É crucial ressaltar que, no âmbito deste projeto, optou-se pela arquitetura *LSTM*. Essa seleção específica foi feita com o intuito de aproveitar a capacidade singular dessas redes neurais em lidar com dependências temporais de longo alcance. Tal característica torna as *LSTMs* altamente eficazes em tarefas complexas de *PLN*, onde a compreensão de relações temporais é essencial. Além disso, é importante destacar que essa abordagem apresenta uma eficiência computacional notável em comparação com técnicas mais recentes, representando assim uma escolha estratégica que equilibra desempenho e custo computacional para atender às demandas do projeto.

A vanguarda do *Deep Learning* no *PLN* é um tema atual e relevante, explorando modelos avançados de aprendizado profundo para abordar desafios complexos, como tra-

dução, resumo, análise de sentimentos e geração de texto. Segundo o DSA (2023), os transformadores representam uma arquitetura de *Deep Learning* inovadora, focada em tarefas *sequence-to-sequence* e capaz de lidar com dependências de longo alcance com facilidade. Baseados integralmente na auto-atenção (Self-Attention), os transformadores computam representações de entrada e saída sem depender de *RNNs* ou convoluções. Introduzidos em 2017 no artigo "Attention is All You Need,"os transformadores substituíram as *RNNs*, oferecendo maior eficiência de treinamento devido à capacidade de lidar com dados sequenciais sem a necessidade de processamento ordenado. Modelos como BERT, GPT-3, T5 e XLNet são exemplos notáveis baseados nessa arquitetura.

2.3.1 Redes Neurais LSTM

A *LSTM (Long Short Term Memory)* é uma *RNN*. As *RNNs* são especialmente projetadas para lidar com problemas nos quais uma amostra não é isolada; em outras palavras, onde o estado ou contexto é de suma importância. Essa característica se encaixa de maneira eficaz em problemas que envolvem tempo, como reconhecimento de fala, modelagem de linguagem, tradução e legendagem de imagens (OLAH, 2015).

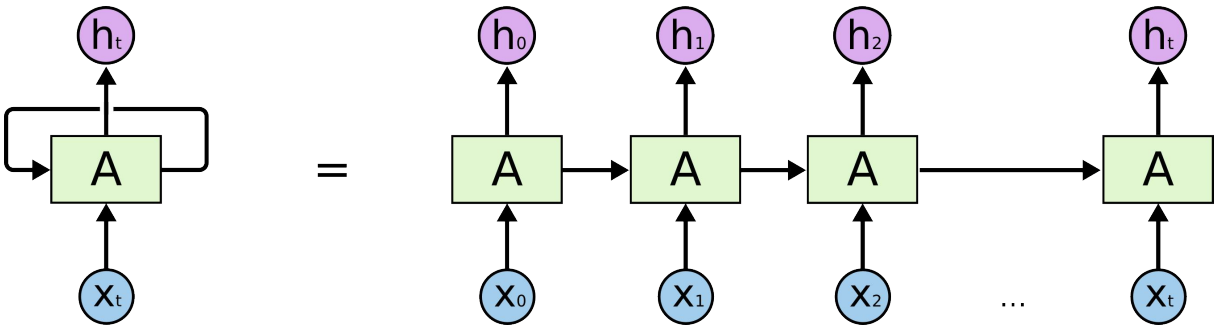

Figura 3 – Arquitetura de *RNN*

Fonte: Olah (2015)

A Figura 3 ilustra a arquitetura de uma *RNN*. Como apresentado, é possível notar que a saída de uma unidade (representada pelos neurônios da *RNN*) serve como entrada para outra, proporcionando, assim, a recorrência (OLAH, 2015). No entanto, apesar de suas vantagens, as *RNNs* enfrentam desafios significativos em relação à memória de longo prazo (OLAH, 2015). Para superar essa limitação, surgem as *LSTMs*, que são capazes de lidar melhor com informações de longo prazo em comparação com as *RNNs* tradicionais.

A capacidade da *LSTM* em lidar com o problema mencionado é explicada por sua arquitetura, conforme ilustrado na Figura 4. Nela, é possível observar uma diferença crucial: a *LSTM* possui duas entradas. Uma delas é a saída da unidade anterior, como em uma *RNN* convencional, enquanto a outra é uma entrada especial que representa o passado.

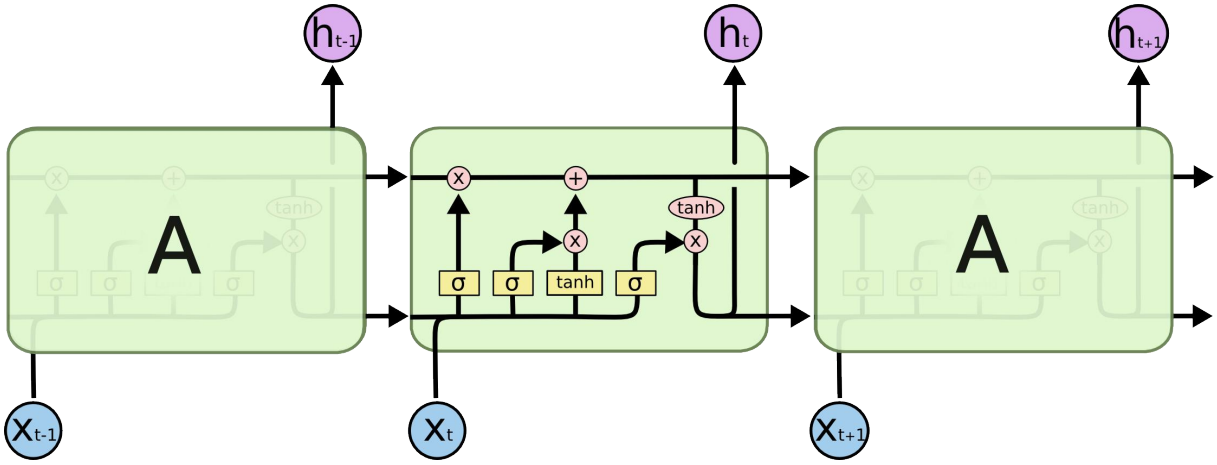

Figura 4 – Arquitetura de *LSTM*

Fonte: Olah (2015)

Essa dualidade de entradas permite que a *LSTM* mantenha e manipule informações ao longo do tempo de uma maneira mais eficaz do que as *RNNs* tradicionais. A capacidade de capturar relações temporais complexas e reter informações relevantes de longo prazo faz da *LSTM* uma escolha poderosa em tarefas que envolvem dependências temporais mais extensas, como reconhecimento de padrões em sequências temporais.

Além disso, um conceito crucial são os *gates* (portas), representados pelos círculos rosa na Figura 4. Eles têm a função de adicionar ou remover informações na *LSTM*, e são conectados a neurônios, sendo três deles com a função sigmoide σ e um com a tangente hiperbólica *tanh*. Esses mecanismos de *gates* permitem que a *LSTM* controle o fluxo de informações, melhorando sua capacidade de aprendizado e retenção de longo prazo(OLAH, 2015).

2.4 Trabalhos Correlatos

O trabalho realizado por Guarise e Rezende (2019), descreve o uso de aprendizado profundo para detectar notícias falsas na internet. O autor explora o uso de um modelo de aprendizado profundo chamado *Hierarchical Attention Networks* (HAN), que usa a estrutura hierárquica dos textos (palavras - frases - documentos) e um mecanismo de atenção que dará peso às partes mais importantes do texto por meio da estrutura arquitetural do *deep learning*. O modelo também permite a visualização dos dados de saída através de um mapa de calor que destaca as palavras e frases mais relevantes para a classificação. O modelo usa modelos pré-treinados para a criação das palavras vetorizadas, usando o algoritmo *Word2Vec*. A base de dados aplicada ao modelo proposto foi a Fake.Br, um corpus contendo notícias da língua portuguesa do Brasil (SILVA et al., 2020). Essa base de dados reúne 7200 notícias sendo elas falsas ou verdadeiras. O modelo de *RNA* implementado foi o HAN, que é baseado em *LSTM*, e dividiu a base de dados em 80% para treinamento e o

restante para avaliar o modelo. Os resultados obtidos do modelo foram satisfatórios, com o valor da *accuracy* de 95.35%, o que é um ótimo resultado se comparando com o mesmo modelo utilizado para uma base de dados da língua inglesa que apresentou 95.4%.

O trabalho de Li et al. (2022) descreve uma rede *deep learning* semi-supervisionada que treina tarefas supervisionadas e não supervisionadas simultaneamente para detectar notícias falsas nas redes sociais e comparar os resultados com alguns métodos de aprendizagem supervisionada, tais como: *Support Vector Machine, Naive Bayes, Bi-LSTM (Bi-directional LSTM) e CNN* . As base de dados empregadas foram a PolitiFact e o conjunto de dados GossipCop do repositório FakeNewsNet, o qual contém notícias falsas sobre atualidades, contexto social e informações espaço-temporais. O desempenho da metodologia proposta foi melhor quando comparada com os outros métodos de aprendizado de máquina.

A tese de doutorado **"Detecção automática de notícias falsas em português"** (SANTOS, 2022) apresentou o corpus na língua português do Brasil, denominado Fake.Br. Na metodologia proposta, destacou três abordagens textuais para análise de uma notícia: abordagens baseadas em atributos linguísticos; abordagens baseadas em conteúdo e abordagens baseadas na estrutura do texto. Para a validação e classificação das notícias, aplicou diferentes técnicas de AM sob cada uma das abordagens: ávores de decisão, Naive Bayes, *Randon Forest*, *K-Nearest Neighbors* (KNN), *Support Vector Machines* (SVM) e redes neurais profundas, com as arquiteturas *Long Short-Term Memory* (LSTM) e *Bidirectional Encoder Representations from Transformers* (BERT).

3 Desenvolvimento

Neste capítulo será descrito as etapas do modelo computacional proposto para a detecção de notícias falsas, bem como, o detalhamento do corpus e sua subdivisão para testar as arquiteturas de *deep learning* implementadas no modelo.

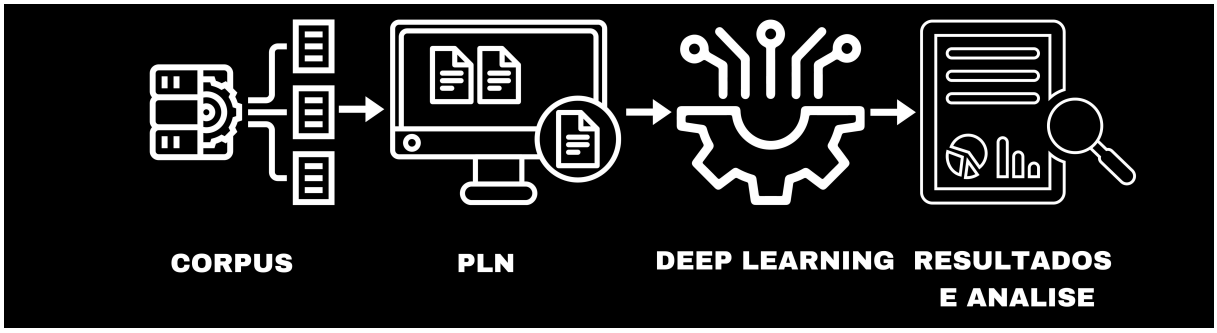

Figura 5 – Etapas do modelo computacional proposto.

3.1 Descrição do Corpus

A coleta de dados ou a escolha da base de dados é o primeiro passo para a aplicação de uma *RNA* . As *RNAs* usam uma quantidade massiva de dados para aprender o padrão através da indução, logo a qualidade e quantidade dos dados usados no treinamento influenciam diretamente no desempenho do modelo. As *RNAs* não são capazes de gerar um conhecimento novo, então tudo o que ela prever ou classificar é derivado dos dados de entrada.

Para o presente trabalho, o conjunto de dados deve incluir tanto notícias verdadeiras quanto falsas, além do corpus ser do idioma português do Brasil. Esses dados podem ser obtidos a partir de fontes confiáveis, como sites de notícias estabelecidos e bancos de dados públicos. O corpus utilizado foi o Corpus Fake.Br, que está disponível em Monteiro et al. (2018). A Figura 6 mostra que o corpus já está balanceado, tendo um total de 7200 notícias classificadas. Dentre estas, 3600 são noticias falsas, as quais serão rotuladas por 0, e 3600 são noticias verdadeiras, rotuladas por 1. O fato do corpus já está balanceado evita alguns problemas e facilita a manipulação.

As notícias falsas foram buscadas manualmente na web sob notícias publicadas em um intervalo de tempo de 2 anos, de janeiro de 2016 a janeiro de 2018, e identificadas em quatro sites: Diário do Brasil, A Folha do Brasil, The Jornal Brasil e Top Five TV. Na sequência, as notícias verdadeiras foram coletadas de forma semi-automática, pois

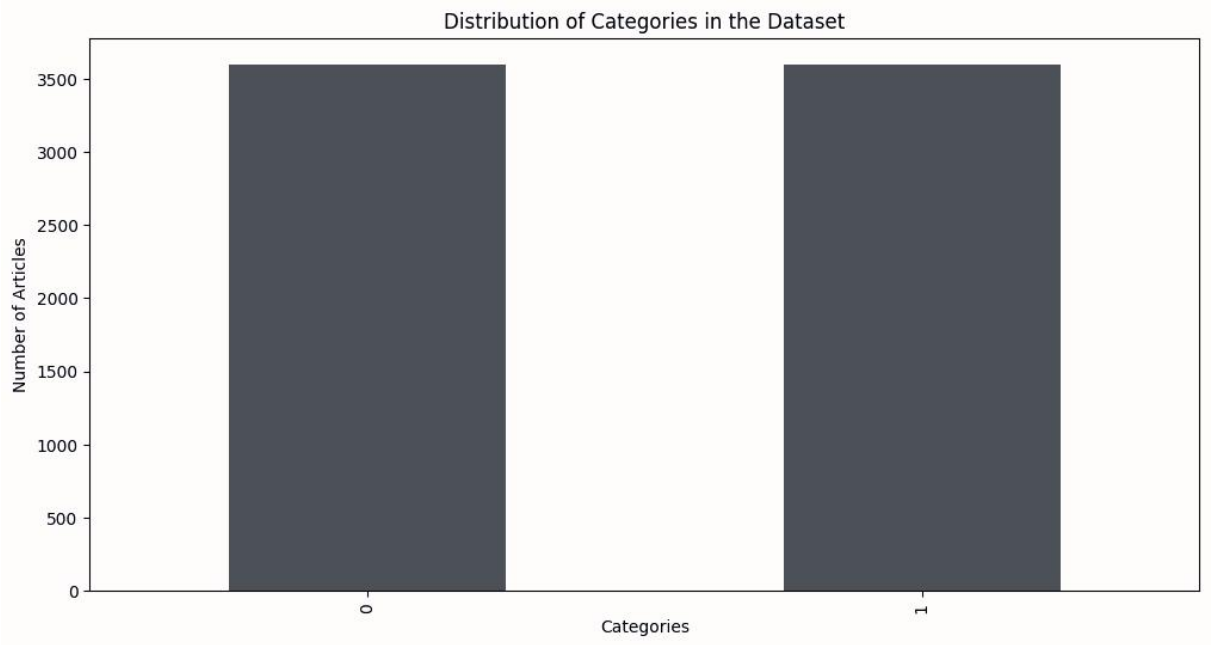

Figura 6 – Distribuição de noticia verdadeiras e falsas

utilizou-se um *web crawler* para coletar as notícias de algumas das principais agências de notícias do Brasil (G1 , Folha de São Paulo e Estadão) (SANTOS, 2022).

3.2 Pré-Processamento da Base de Dados

Quando as *RNAs* são utilizados com dados textuais, os dados de entrada para a rede precisão ser pre-processados. Neste momento as técnicas de *PLN* são empregadas sob os dados textuais realizando a limpeza de dados desnecessários e a transformação de palavras em números por algoritmos como o *word embedding*.

A limpeza dos dados representa a retirada de pontuações, números, acentos, símbolos diversos e palavras comuns, tais como artigos e preposições da língua portuguesa. Além disso, converteu palavras maiúsculas em minúsculas, aplicou lematização e tokenização. Ao final destes processo, cada noticia é transformada em uma lista de palavras, denominadas tokens.

Cada token precisa ser representado por um número para servir de dado de entrada para a *RNA*. Existem vários tipo de representação possíveis, neste estudo a utilizada foi o algoritmo *Word2Vec*.

Como o objetivo do modelo é classificar noticias falsas, logo os dados de entrada são noticias, e cada noticia foi transformada numa lista de tokens. Com o auxílio de um dicionario de tokens, codifica-se as listas de tokens e as transforma em uma lista de *word embedding*.

O trabalho foi inteiramente desenvolvido na IDE Notebook *Python*, e as principais bibliotecas utilizadas para o Processamento de Linguagem Natural foram o *pandas, numpy* e *spacy*. Para a criação das listas *word embedding* a biblioteca escolhida foi *gensim*.

3.3 Modelo computacional

Existem várias arquiteturas de *RNAs* que podem ser utilizadas para problemas direcionados ao processamento de linguagem natural. Este trabalho explorou as *RNAs* dos tipos *LSTMs*, as quais foram implementadas na linguagem de programação *Python* pela biblioteca *Tensorflow*.

3.3.1 Separação dos Dados

A base de dados foi dividida nas três partes treino, validação e teste, e em várias percentagens, com mostra a Tabela 1. Essa separação foi feita através da função *train_test_split* da biblioteca *sklearn*, utilizando como parâmetros os valores citados da Tabela 1.

Divisão do Dataset(%)								
Treino	50	55	60	65	70	75	80	85
Validação	25	22.5	20	17.5	15	12.5	10	7.5
Teste	25	22.5	20	17.5	15	12.5	10	7.5

Tabela 1 – Subdivisão da base de dados para treino, teste e validação do modelo proposto.

3.3.2 Arquiteturas das RNA

Foi utilizado três *RNAs* baseado na arquitetura da *LSTM*, que são a própria *LSTM, Stacked LSTM e Bi-LSTM*. A Figura 7 mostra, respectivamente, as arquiteturas dos modelos.

A primeira arquitetura da Figura 7 representa a *RNA LSTM* padrão. Ela é composta por quatro camadas. A primeira camada, ou a camada de entrada é um vetor de 992 posição, onde cada posição é um numero que representa a palavra da notícia. Esses números são interpretados como chaves e que serão utilizados como referencia no dicionario do *Word2Vec*, pois cada token é traduzido por um vetor de 100 posição. Assim, a saída da segunda camada é uma matriz de dimensão 992x100. As duas próximas camadas, denominadas de densas, são constituídas de 128 neurônicos, o qual resultou num vetor de 128 posições que será a entrada da última camada. A função de ativação da última camada é a *softmax*, que com apenas dois neurônios, resultará em dois valores possíveis: 0, para notícia verdadeira, ou 1 para notícia falsa.

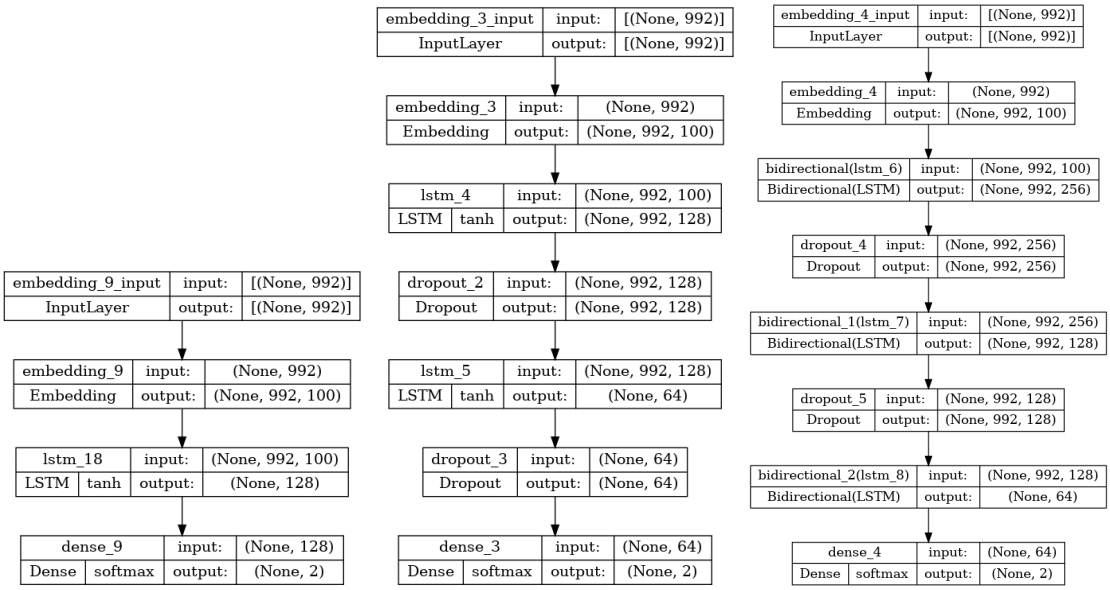

Figura 7 – Arquitetura das *RNAs*.

Já a segunda arquitetura da Figura 7, representa a *RNA Stacked LSTM*. As primeiras camadas da *RNA LSTM* e a última são iguais para as outras duas arquiteturas, mudando apenas a quantidade de camadas escondidas. Na *RNA Stacked LSTM* tem o total de quatro camadas escondidas, que são duas *LSTM* e duas Dropout intercaladas entre si. A Dropout, quando adicionada, ela gera uma probabilidade p de cortar a conexão de um neurônio para os neurônios das *LSTMs*, com o intuito de partir de uma *RNA* densa para uma *RNA diluída* (GYLBERTH, 2019). Essa técnica ajuda a controlar overfitting. Neste modelo foi usado $p = 0.2$ para os dois Dropout, uma *LSTM* com 128 neurônios e a outra com 64 neurônios.

A terceira arquitetura da Figura 7 *Bi-LSTM* a , que é formada por cinco camadas escondidas, sendo três *LSTM* bidirecional e duas Dropout com $p = 0.5$. A principal diferença entre as duas arquiteturas anteriores é que a camada *Bi-LSTM* é composta por duas *LSTMs*, sendo que uma vai do passado para o futuro e a outra vai do futuro para o passado, o que faz essa *RNA* extrair características mais complexas. As *RNAs LSTM* bidirecional possuem 256, 128 e 64 neurônios.

Um parâmetro importante no treinamento de uma *RNA* é o número de épocas de um modelo. Um número baixo de épocas pode causar o problema conhecido como *underfitting*, que significa que os pesos não se adaptaram o suficiente ao conjunto de dados. Já com um número alto de épocas pode gerar outro problema que é o *overfitting*, que significa que os pesos se especializaram no conjunto de treino, tirando o poder de generalização do modelo. Isto faz que o modelo seja especialista no conjunto de dados treinados, mas ao inserir novos dados o modelo não irá ter um bom resultado. Desta forma, o objetivo é encontrar o número de épocas entre os dois problemas.

A escolha do melhor valor para época é uma tarefa empírica, e para minimizar esse problema, este projeto tem uma função de *callbacks*. Na etapa de treinamento do modelo são passados dois conjuntos de dados como entrada, um chamado treino e outro chamado validação. A função *callback* pegará os pesos resultados de cada época e validará no conjunto de validação, obtendo assim algumas estatísticas importantes como o a função *loss*. Com essas estatísticas, ele irá fazer uma timeline de todas as épocas aponto que no momento em que as épocas pararem de melhorar e a função *loss* parar de diminuir, o treino é interrompido e é devolvido os melhores pesos obtidos nas épocas passadas. A função *callbacks* foi implementada com a função EarlyStopping da bibloteca *Tensorflow*.

Foram treinados para as três arquiteturas propostas oito subdivisões diferentes da base de dados, com porcentagem variando de 50% a 85%, com um intervalo de 5% dados para treinamento, conforme mostra a Tabela 1. Essa abordagem foi escolhida por tentativa e erro. Cada modelo levou quatro horas em média para concluir o treinamento.

4 Resultados

O procedimento para avaliar o desempenho dos modelos de *RNAs* na classificação de notícias falsas deriva da aplicação do conjunto de testes da base de dados aos modelos computacionais treinados e validados. Os resultados gerados pelo conjunto de teste são exibidos na matriz de confusão, uma ferramenta valiosa para analisar o desempenho de modelos de classificação. A representação genérica dessa matriz é ilustrada na Figura 8.

		Classe Prevista	
		classe 0 (FAKENEWS)	classe 1
Classe Verdadeira	classe 0 (FAKENEWS)	Verdadeiro 0 (V0)	Falso 1 (F1)
	classe 1	Falso 0 (F0)	Verdadeiro 1 (V1)

Figura 8 – Exemplo de matriz de confusão.

Essas matrizes são empregadas para derivar métricas de avaliação que proporcionam um entendimento mais aprofundado das áreas em que os modelos se destacam e de suas limitações. A Figura 9 apresenta todas as matrizes de confusão em relação à bases de dados de teste.

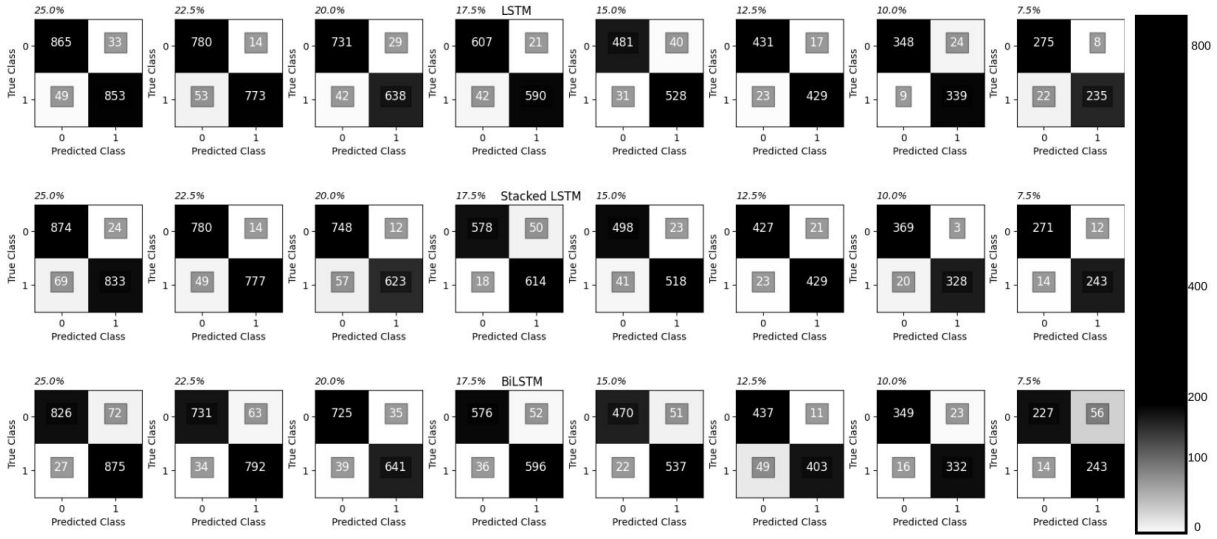

Figura 9 – Matrizes de confusão de todos os modelos aplicados ao conjunto de dados de teste.

A métrica mais abrangente para problemas discretos e frequentemente a mais

intuitiva é a *accuracy*, que representa a porcentagem de acertos de um modelo. O cálculo da *accuracy* pelas variáveis apresentadas na matriz de confusão da Figura 8, são exibidos pela fórmula descrita na Equação 4.1. Em outras palavras, trata-se da probabilidade de um modelo classificar corretamente as amostras.

$$\text{accuracy} = \frac{\sum_{n=0}^{class}(V_n)}{\sum_{n=0}^{class}(V_n + F_n)} \tag{4.1}$$

A Equação 4.1 representa a soma de todas as amostras classificadas corretamente, dividida pelo universo total de amostras. A Tabela 2 exibe a *accuracy* de cada um dos modelos desenvolvidos, com o melhor resultado sendo o *Stacked LSTM* da subdivisão "80,10,10", atingindo 96.80%, e o pior resultado sendo a *Bi-LSTM* da subdivisão "85,7.5,7.5", com 87.03%. A média geral foi de 94.44%. A *LSTM* teve uma média de 95.02%, variando de 93.42% a 95.86%, com uma diferença de $\Delta = 2.44\%$. A *Stacked LSTM* teve uma média de 95.24%, variando de 94.07% a 96.80%, com uma diferença de $\Delta = 2.73\%$. Por fim, a *Bi-LSTM* teve uma média de 93.07%, variando de 87.03% até 94.86%, com uma diferença de $\Delta = 7.83\%$.

Accuracy			
%	LSTM	Stacked LSTM	Bi-LSTM
50,25,25	95.44	94.83	94.49
55,22.5,22.5	95.86	96.11	94.01
60,20,20	95.06	95.20	94.86
65,17.5,17.5	94.99	94.60	93.01
70,15,15	93.42	94.07	93.24
75,12.5,12.5	95.55	95.11	93.33
80,10,10	95.41	96.80	94.58
85,7.5,7.5	94.44	95.18	87.03

Tabela 2 – Tabela de Resultados: *Accuracy*.

A *accuracy* é uma métrica útil, mas oferece apenas uma visão parcial dos resultados, o que não garante necessariamente que o modelo é eficaz, especialmente em casos de desbalanceamento do conjunto de dados. Por isso, outras métricas, como *precision*, são complementares. Alguns autores a chamam de Valor Preditivo Positivo (VPP) ou Valor Preditivo Negativo (VPN) em problemas de classificação binária, isto é, é a *precision* para a classe 1 e a *precision* para classe 0. O significado dessa métrica significa que se classificou uma amostra como $Class_a$ qual é a probabilidade da classificação está correta. Usando a matriz a Figura 8, a Equação 4.2 mostra as variáveis envolvidas no cálculo desta métrica.

$$precision = \frac{V_{class}}{V_{class} + F_{class}} \tag{4.2}$$

A Tabela 3 mostra a *precision* de cada um dos modelos, tendo como melhor resultado a *RNA Stacked LSTM* na subdivisão "80,10,10"para Classe 1 com 99.09%, e o pior foi a rede *Bi-LSTM* na subdivisão "85,7.5,7.5"para Classe 1 com 81.27%. O melhor e o pior para Classe 0 foi *LSTM* da subdivisão "80,10,10"com 97.48% e a *Bi-LSTM* da subdivisão "80,10,10"com 89.96%. A média geral é de 94.55%, mas para a Classe 0 é de 94.41% e para a Classe 1 é de 94.70%.

O modelo com as redes *LSTM* obteve para a classe 0 uma média de 94.42% que variou de 92.59% até 97.48%, com um $\Delta = 4.89\%$, e para a Classe 1 uma média de 95.75% que variou de 92.96% até 98.22%, resultando um $\Delta = 5.26\%$. O modelo da *RNA Stacked LSTM* obteve para a classe 0 uma média de 94.24% que variou de 92.39% até 96.98% com um $\Delta = 4.59\%$ e para a Classe 1 uma média de 95.29% que variou de 92.47% até 99.09% com um $\Delta = 6.62\%$. Por fim, o modelo com as redes *Bi-LSTM* obteve da classe 0 uma média de 94.58% que variou de 89.92% até 96.83%, com um $\Delta = 6.91\%$ e para a Classe 1 uma média de 91.91% que variou de 81.27% até 97.34% com um $\Delta = 16.07\%$.

Precision						
	LSTM		Stacked LSTM		Bi-LSTM	
%	Classe 0	Classe 1	Classe 0	Classe 1	Classe 0	Classe 1
50,25,25	94.64	96.28	92.68	97.20	96.83	92.40
55,22.5,22.5	93.64	98.22	94.09	98.23	95.56	92.63
60,20,20	94.57	95.65	92.92	98.11	94.90	94.82
65,17.5,17.5	93.53	96.56	96.98	92.47	94.12	91.98
70,15,15	93.95	92.96	92.39	95.75	95.53	91.33
75,12.5,12.5	94.93	96.19	94.89	95.33	89.92	97.34
80,10,10	97.48	93.39	94.86	99.09	95.62	93.52
85,7.5,7.5	92.59	96.71	95.09	95.29	94.19	81.27

Tabela 3 – Tabela de Resultados: *Precision*.

Como mostrado, cada modelo apresenta dois valores da *precision*, referentes ao percentual de acertos para cada classe classificada. Isso implica que a *precision* de uma classe pode ser substancialmente diferente da *precision* da outra classe, uma particularidade que não é perceptível com a métrica *accuracy*.

A Tabela 4 foi construída com base na Equação 4.3, exibindo o valor de Δ da métrica *precision* para cada modelo. Um Δ*precision* maior indica que o modelo tem uma tendência a acertar a classificação de uma subdivisão dos conjuntos de dados específica, enquanto valores menores indicam maior similaridade de acertos entre todas subdivisões do conjunto de dados testados. Embora o ideal seja uma *precision* de 100%, resultando em Δ*precision* $= 0$, essa perfeição nem sempre é alcançável. A escolha entre um Δ*precision* alto ou baixo dependerá muito do problema em questão. Em alguns casos, ter um alto grau de certeza para uma classe pode ser mais crucial do que em outros.

$$\Delta precision = |(Class0)_{i,j} - (Class1)_{i,j}| \tag{4.3}$$

A Tabela 4 mostra, que a *Bi-LSTM* na subdivisão "60,20,20"teve a menor diferença entre os modelos (0.08%) e a *Bi-LSTM* na subdivisão "85,7.5,7.5"teve a maior diferença $\Delta = 12.84\%$ e uma média geral de 3.48%. As redes *LSTM* teve uma média de 2.60% e variou de 0.99% até 4.58% com um $\Delta = 3.59\%$, a *Stacked LSTM* teve uma média de 4.53% e variou de 0.20% até 5.19% com um $\Delta = 4.99\%$ e a *Bi-LSTM* teve uma média 2.60%.

Modulo da Diferença da Precision			
%	LSTM	Stacked LSTM	Bi-LSTM
50,25,25	1.64	4.52	4.43
55,22.5,22.5	4.58	4.14	2.93
60,20,20	1.08	5.19	0.08
65,17.5,17.5	3.03	4.51	2.14
70,15,15	0.99	3.36	4.20
75,12.5,12.5	1.26	0.44	7.42
80,10,10	4.09	4.23	2.10
85,7.5,7.5	4.12	0.20	12.92

Tabela 4 – Tabela de Resultados: Modulo da diferença da Classes 0 e 1 de Precision.

O *recall* é outra métrica muito usada, que alguns autores, para problemas de classificação binária, chamam o *recall* para a classe positiva de sensibilidade e para a classe negativa de especificidade. *Recall* é a classe prevista corretamente dividida pelo total da classe. Em outras palavras, dada uma amostra de uma $Class_a$, qual é a probabilidade da classificação do modelo ser $Class_a$.

$$recall = \frac{V_{class}}{class} \tag{4.4}$$

A Tabela 5 mostra o *recall* de cada um dos modelos, tendo como melhor resultado a *Stacked LSTM* na subdivisão "80,10,10"para Classe 0 com 99.19%, e o pior no geral foi *Bi-LSTM* na subdivisão "85,7.5,7.5"para Classe 0 com 80.21%. O melhor e o pior para Classe 1 foi a rede *LSTM* de na subdivisão "80,10,10"com 97.41% e a *Bi-LSTM* na subdivisão "75,12.5,12.5"com 89.16%. A média geral é de 94.44%, para a Classe 0 é de 94.65% e para a Classe 1 é de 94.24%.

Em relação à métrica *recall*, só as redes *LSTM* obteve para a classe 0 uma média de 95.83% que variou de 92.32% até 98.24% com um $\Delta = 5.92\%$, e para a classe 1 uma média de 94.19% que variou de 91.44% até 97.41% com um $\Delta = 5.97\%$. A rede *Stacked LSTM* obteve para a classe 0 uma média de 96.49% que variou de 92.04% até 99.19% com

Recall						
%	LSTM		Stacked LSTM		Bi-LSTM	
	Classe 0	Classe 1	Classe 0	Classe 1	Classe 0	Classe 1
50,25,25	96.33	94.57	97.33	92.35	91.98	97.01
55,22.5,22.5	98.24	93.58	98.24	94.07	92.07	95.88
60,20,20	96.18	93.82	98.42	91.62	95.39	94.26
65,17.5,17.5	96.66	93.35	92.04	97.15	91.72	94.30
70,15,15	92.32	94.45	95.59	92.67	90.21	96.06
75,12.5,12.5	96.21	94.91	95.31	94.91	97.54	89.16
80,10,10	93.55	97.41	99.19	94.25	93.82	95.40
85,7.5,7.5	97.17	91.44	95.76	94.55	80.21	94.55

Tabela 5 – Tabela de Resultados: Recall.

um $\Delta = 7.15\%$, e para a classe 1 uma média de 93.95% que variou de 91.62% até 97.15% com um $\Delta = 5.53\%$. Por fim, a *Bi-LSTM* obteve da classe 0 uma média de 94.62% que variou de 80.21% até 97.54% com um $\Delta = 17.33\%$ e para a classe 1 uma média de 94.58% que variou de 88.16% até 97.01% com um $\Delta = 7.85\%$.

A Tabela 6 é o módulo da diferença da *recall* e mostra, que a *Stacked LSTM* na subdivisão "75,12.5,12.5"obteve a menor diferança entre modelos de 0.40% e *Bi-LSTM* na subdivisão "85,7.5,7.5"obteve a maior diferença com 14.10% e uma média geral de 4.10%. As redes *LSTM* teve uma média de 3.14% e variou de 1.30% até 5.73% com um $\Delta = 4.43\%$, a rede *Stacked LSTM* teve uma média de 3.80% e variou de 0.40% até 5.19% com um $\Delta = 6.40\%$ e a *Bi-LSTM* teve uma média 5.34% e variou de 1.13% até 14.34% um $\Delta = 13.21\%$.

Módulo da Diferença do Recall			
%	LSTM	Stacked LSTM	Bi-LSTM
50,25,25	1.76	4.98	5.03
55,22.5,22.5	4.66	4.17	3.81
60,20,20	2.36	6.80	1.13
65,17.5,17.5	3.31	5.11	2.58
70,15,15	2.13	2.92	5.85
75,12.5,12.5	1.30	0.40	8.38
80,10,10	3.86	4.94	1.58
85,7.5,7.5	5.73	1.21	14.34

Tabela 6 – Tabela de Resultados: Módulo da diferença da Classes 0 e 1 do Recall.

O *F1-Score* é outra métrica, definida pela média harmônica da *precision* da $Class_n$ e do *recall* $Class_n$. Ela é particularmente útil quando a base de dados está desbalanceada. Sua fórmula está descrita na Equação 4.5.

$$F_1score = \frac{2 * precision(class) * recall(class)}{precision(class) + recall(class)} \tag{4.5}$$

A Tabela 7 mostra o *F1-Score* de cada um dos modelos, tendo como melhor resultado a rede *Stacked LSTM* na subdivisão "80,10,10"para classe 0 com 96.98%, e o pior a rede *Bi-LSTM* na subdivisão "85,7.5,7.5"para classe 0 com 86.64%. O melhor e o pior para classe 1 foi *Stacked LSTM* na subdivisão "80,10,10"com 96.61% e a *Bi-LSTM* na subdivisão "85,7.5,7.5"com 87.41%. A média geral é de 94.44%. O melhor e o pior para classe 0 é de 94.47% e para a Classe 1 é de 94.41%.

As redes *LSTM* teve para a classe 0 uma média de 95.10% que variou de 93.13% até 95.88%, com um $\Delta = 2.75\%$ e para a classe 1 uma média de 94.94% que variou de 93.70% até 95.85% com um $\Delta = 2.15\%$. A rede *Stacked LSTM* obteve para a classe 0 uma média de 95.32% que variou de 93.96% até 96.98% com um $\Delta = 3.02\%$ e para a classe 1 uma média de 95.14% que variou de 94.18% até 96.61% com um de $\Delta = 2.43\%$. Por fim a *Bi-LSTM* obteve para a classe 0 uma média de 92.99% que variou de 86.64% até 95.14% com um $\Delta = 8.50\%$ e para a classe 1 uma média de 93.14% que variou de 87.41% até 94.65% com um $\Delta = 7.24\%$.

F1-Score						
	LSTM		Stacked LSTM		Bi-LSTM	
%	Classe 0	Classe 1	Classe 0	Classe 1	Classe 0	Classe 1
50,25,25	95.47	95.41	94.95	94.71	94.35	94.65
55,22.5,22.5	95.88	95.85	96.12	96.10	93.78	94.23
60,20,20	95.37	94.73	95.59	94.75	95.14	94.54
65,17.5,17.5	95.07	94.93	94.44	94.75	92.90	93.12
70,15,15	93.13	93.70	93.96	94.18	92.79	93.64
75,12.5,12.5	95.57	95.55	95.10	95.12	93.58	93.07
80,10,10	95.47	95.36	96.98	96.61	94.71	94.45
85,7.5,7.5	94.83	94.00	95.42	94.92	86.64	87.41

Tabela 7 – Tabela de Resultados: *F1-Score*.

Como mostrado, o *F1-Score* é uma métrica específica para uma classe. Existem outras métricas derivadas do *F1-Score* que proporcionam uma visão mais geral de cada modelo. O *Average F1-Score* é a média aritmética dos *F1-Scores* de cada classe, e o seu cálculo está descrito na Equação 4.6.

$$AverageF_1score = \frac{\Sigma_{n=0}^{Class} F_1score(n)}{class} \tag{4.6}$$

A Tabela 8 mostra que o melhor resultado foi obtido pela rede *Stacked LSTM* na subdvisão "80,10,10"com 96.80% e o pior resultado pela rede *Bi-LSTM* na subdvisão "85,7.5,7.5"com 87.03% e a média geral de 94,44%. As redes *LSTM* teve uma média de

Average F1-Score			
%	LSTM	Stacked LSTM	Bi-LSTM
50,25,25	95.44	94.83	94.50
55,22.5,22.5	95.86	96.11	94.00
60,20,20	95.05	95.17	94.84
65,17.5,17.5	95.00	94.60	93.01
70,15,15	93.41	94.07	93.21
75,12.5,12.5	95.56	95.11	93.32
80,10,10	95.42	96.80	94.58
85,7.5,7.5	94.41	95.17	87.03

Tabela 8 – Tabela de Resultados: *Average F1-Score*.

95.02% e variou de 93.42% até 95.86% com um $\Delta = 2.44\%$, a *Stacked LSTM* teve uma média de 95.24% e variou de 94.07% até 96.80% com um $\Delta = 2.73\%$. Por fim, a rede *Bi-LSTM* teve uma média de 93,07% variou de 87.03% até 94.86 com um $\Delta = 7.83\%$. A *average F1-Score* e a *accuracy* apresentam os mesmos valores devido ao balanceamento da base de dados.

4.0.1 Comparações com Outros Trabalhos

A Tabela 9 apresenta os resultados obtidos pelos modelos propostos por Monteiro et al. (2018) e Guarise e Rezende (2019), comparativamente à Stacked LSTM 80,10,10 desenvolvida neste trabalho. Todos os modelos foram avaliados utilizando o mesmo conjunto de dados do Fake.br (MONTEIRO et al., 2018). Em relação aos trabalhos anteriormente citados, destaca-se que o modelo desenvolvido neste trabalho demonstrou um desempenho superior em todas as métricas avaliadas. Essa melhoria significativa reforça a eficácia da abordagem baseada em Stacked LSTM 80,10,10 para a detecção de informações falsas, consolidando sua posição como uma contribuição relevante no campo de estudos sobre detecção de notícias falsas.

Tabela 9 – Comparação dos resultados encontrados com em Monteiro et al. (2018) e Guarise e Rezende (2019)

	Precision		Recall		Accuracy
	Classe 0	Classe 1	Classe 0	Classe 1	
Monteiro et al. (2018)	0.89	0.89	0.89	0.89	0.89
Guarise e Rezende (2019)	0.97	0.94	0.94	0.97	0.95
Stacked LSTM 80,10,10	0.95	0.99	0.99	0.94	0.97

5 Conclusão

O objetivo deste trabalho foi construir modelos computacionais para a classificação de notícias falsas para o português do Brasil, utilizando técnicas de *PLN* aliadas às *RNAs deep learning*. Esses modelos foram processados no corpus Fake.Br (MONTEIRO et al., 2018) e construídos sob os algoritmos *Word2Vec*, juntamente com as arquiteturas das *RNAs LSTM* , *Stacked LSTM* e *Bi-LSTM*.

Ao avaliar o desempenho dos modelos das *RNAs* na classificação, obteve uma visão abrangente de seus comportamentos em diferentes configurações dos conjuntos de dados de teste, os quais foram analisados pelas métricas *accuracy*, *precision, recall* e *F1-Score*. Os resultados indicam que, em termos de *accuracy*, o modelo *Stacked LSTM* na divisão de 80% para treino, 10% para validação e 10% para teste obteve o melhor desempenho, atingindo 96.80%. No entanto, é crucial considerar outras métricas além da *accuracy* para obter percepções mais detalhadas sobre o comportamento dos modelos.

Analisando a *precision, recall* e *F1-Score*, percebe-se que diferentes modelos apresentam diferentes pontos fortes e fracos ao relacionar e comparar às classes 0 e 1. A rede *Bi-LSTM*, em média, demonstrou uma diferença nos desempenhos entre as classes, como por exemplo, a *precision* obteve um valor menor para a classe 1 em comparação com a classe 0. Esse desequilíbrio destaca a importância de considerar métricas complementares para avaliar a eficácia dos modelos propostos.

Outro modelo que chamou a atenção por sua consistência foi a rede *Bi-LSTM* no conjunto de dados subdividido em 60% para treino, 20% para validação e 20%. Este modelo obteve uma *accuracy* de 95.08%, além de que sua *precision*, para ambas as classes, apresentaram uma variação de 0.08%.

Além disso, ao avaliar a média geral das métricas, observa-se que todos os modelos tiveram um desempenho bastante sólido, com médias superiores a 90% em todas as métricas analisadas. No entanto, é essencial adaptar o modelo e suas configurações com base nas necessidades e nas características específicas do problema em questão.

O trabalho realizado na detecção de notícias falsas em português do Brasil é fundamental para fortalecer a capacidade de combate à desinformação. Os modelos de classificação desenvolvidos neste trabalho oferecem uma contribuição direta à promoção da informação precisa e confiável para a sociedade. Governos, organizações e instituições de ensino podem aproveitar os modelos aqui sugeridos e implementar sistemas computacionais para detecção de notícias falsas em plataformas online e em redes sociais. Isto facilitaria a tomada de decisões, especialmente em momentos críticos, prevenindo a propagação de informações falsas, as quais prejudicariam a estabilidade social. Este esforço

conjunto representa um passo significativo na construção de um ambiente de informação mais seguro e confiável para a sociedade brasileira.

As técnicas e percepções desenvolvidos não se limitam apenas à detecção de notícias falsas, mas podem ser aplicados em outras áreas que envolvam processamento de linguagem natural e classificação de texto, ampliando assim o impacto potencial do trabalho para além do domínio específico abordado. Como por exemplo, análise de sentimentos, classificação automática de texto, pesquisa e recomendação de Conteúdo, extração e reconhecimento de entidades.

Este trabalho enfrenta algumas limitações importantes que merecem consideração. Em primeiro lugar, a base de dados utilizada é relativamente pequena, o que pode impactar a generalização dos modelos para um ambiente real. É crucial reconhecer que a diversidade e a complexidade das notícias falsas podem não estar completamente representadas em uma base de dados limitada, o que pode afetar a eficácia do modelo em situações do mundo real.

Outra limitação significativa está relacionada ao tempo e aos recursos computacionais. O desenvolvimento e treinamento de modelos de detecção de notícias falsas exigem consideráveis recursos computacionais, e uma base de dados maior pode aumentar ainda mais essas demandas. Isso não apenas impacta a viabilidade prática da implementação em larga escala, mas também destaca a necessidade de investimentos adicionais em infraestrutura computacional para lidar com conjuntos de dados mais extensos.

Para trabalhos futuros, pretende-se investigar estratégias de pré-treinamento para o contexto linguístico brasileiro, a fim de otimizar o desempenho do modelo em relação às características únicas da língua portuguesa. Além disso, propõem-se buscar novas arquiteturas de *RNA*, especialmente as baseadas em transformers, como o BERT (Bidirectional Encoder Representations from Transformers) ou GPT (Generative Pre-trained Transformer), devido a capacidade destas em capturar relações semânticas mais complexas e em contextos mais amplos.

Essa abordagem visa não apenas melhorar a eficácia na detecção de notícias falsas, mas também contribuir para o avanço contínuo da pesquisa em processamento de linguagem natural e na mitigação dos desafios associados à desinformação. Estou entusiasmado com a oportunidade de explorar essas novas perspectivas e aprimorar ainda mais a capacidade de combater a disseminação de informações falsas na sociedade.

Referências

ALVES, M. A. S.; MACIEL, E. R. H. O fenômeno das fake news: definição, combate e contexto. **Internet & sociedade**, Universidade Federal de Minas Gerais, 2020. Citado na página 14.

BIRD, S.; KLEIN, E.; LOPER, E. **Natural language processing with Python: analyzing text with the natural language toolkit**. [S.l.]: "O'Reilly Media, Inc.", 2009. Citado na página 14.

CHAGAS, E. T. D. O. Deep learning e suas aplicações na atualidade. **Revista Científica Multidisciplinar Núcleo do Conhecimento**, v. 04, n. 05, p. 05–26, Maio 2019. ISSN 2448-0959. Disponível em: <https://www.nucleodoconhecimento.com.br/administracao/deep-learning>. Citado na página 17.

DSA. **Deep Learning Book**. 2023. Disponível em: <https://www.deeplearningbook.com.br/transformadores-o-estado-da-arte-em-processamento-de-linguagem-natural/>. Acesso em: 29 de Maio 2023. Citado na página 18.

EDRONE. **O que é Word2Vec?** 2023. Acessado em 2023-11-24. Disponível em: <https://edrone.me/pt/blog/o-que-e-word2vec>. Citado na página 15.

FORMIGA, I. C. **Análise de Sentimentos em Português utilizando Machine Learning**. 2022. Built by Oinkina with Hakyll. Accessed on 10 de janeiro de 2024. Disponível em: <https://medium.com/@igorformiga125/an%C3%A1lise-de-sentimentos-em-portugu%C3%AAs-utilizando-machine-learning-94a20fff8bef>. Citado na página 10.

GUARISE, L.; REZENDE, S. O. **Detecção de notícias falsas usando técnicas de deep learning**. Dissertação (Mestrado) — Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, 2019. Citado 3 vezes nas páginas 7, 19 e 32.

GYLBERTH, R. **Understanding Dropout**. 2019. Published in Konvergen.AI. Accessed on 10 de janeiro de 2024. Disponível em: <https://medium.com/konvergen/understanding-dropout-ddb60c9f98aa>. Citado na página 24.

IBM Brasil. **Deep Learning**. 2023. Disponível em: <https://www.ibm.com/br-pt/cloud/deep-learning>. Acesso em: 29 de Maio 2023. Citado 2 vezes nas páginas 10 e 17.

LI, X.; LU, P.; HU, L.; WANG, X.; LU, L. A novel self-learning semi-supervised deep learning network to detect fake news on social media. **Multimedia Tools and Applications**, v. 81, n. 14, p. 19341 – 19349, June 2022. ISSN 1573-7721. Disponível em: <https://doi.org/10.1007/s11042-021-11065-x>. Citado na página 20.

MCCORMICK, C. **Word2Vec Tutorial - The Skip-Gram Model**. 2016. Retrieved on 2023-11-24. Disponível em: <http://www.mccormickml.com>. Citado na página 16.

MONTEIRO, R. A.; SANTOS, R. L. S.; PARDO, T. A. S.; ALMEIDA, T. A. de; RUIZ, E. E. S.; VALE, O. A. Contributions to the study of fake news in portuguese: New corpus and automatic detection results. In: **Computational Processing of the**

Portuguese Language. [S.l.]: Springer International Publishing, 2018. p. 324–334. ISBN 978-3-319-99722-3. Citado 5 vezes nas páginas 7, 12, 21, 32 e 33.

OGUNDEPO, O. **Understanding Word2Vec**. 2021. Published on Jul 23, 2021. Accessed on 10 de janeiro de 2024. Disponível em: <https://medium.com/analytics-vidhya/understanding-word2vec-39fabe660705>. Citado na página 15.

OLAH, C. **Understanding LSTM Networks**. 2015. Posted on August 27, 2015. Built by Oinkina with Hakyll. Accessed on 10 de janeiro de 2024. Disponível em: <https://colah.github.io/posts/2015-08/Understanding-LSTMs/>. Citado 2 vezes nas páginas 18 e 19.

Oracle. **NLP oracle**. 2023. Disponível em: <https://www.oracle.com/br/artificial-intelligence/what-is-natural-language-processing/>. Acesso em: 29 de Maio 2023. Citado na página 10.

RODRIGUES, T. M.; BONONE, L.; MIELLI, R. Desinformação e crise da democracia no brasil: é possível regular fake news? **Confluências| Revista Interdisciplinar de Sociologia e Direito**, v. 22, n. 3, p. 30–52, 2020. Citado na página 10.

SANTOS, R. L. d. S. **Detecção automática de notícias falsas em português**. Tese (Doutorado) — Universidade de São Paulo, São Carlos, SP, 6 2022. Disponível em: <https://www.teses.usp.br/teses/disponiveis/55/55134/tde-14072022-165613/pt-br.php>. Citado 2 vezes nas páginas 20 e 22.

SILVA, R. M.; SANTOS, R. L.; ALMEIDA, T. A.; PARDO, T. A. Towards automatically filtering fake news in portuguese. **Expert Systems with Applications**, v. 146, p. 113199, 2020. ISSN 0957-4174. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0957417420300257>. Citado na página 19.

TEIXEIRA, A.; SANTOS, R. D. C. et al. Fake news colocam a vida em risco: a polêmica da campanha de vacinação contra a febre amarela no brasil. 2020. Citado na página 10.