

INSTITUTO FEDERAL DO ESPÍRITO SANTO
CURSO SUPERIOR DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

HARÃ HEIQUE DOS SANTOS

**REDES NEURAIS SIAMESAS LSTM PARA DETERMINAÇÃO DE
SIMILARIDADES ENTRE PARES DE SENTENÇAS LITERÁRIAS**

Serra
2021

HARÃ HEIQUE DOS SANTOS

**REDES NEURAIS SIAMESAS LSTM PARA DETERMINAÇÃO DE
SIMILARIDADES ENTRE PARES DE SENTENÇAS LITERÁRIAS**

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Bacharelado em Sistemas de Informação do Instituto Federal do Espírito Santo, Campus Serra, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Fidelis Zanetti de Castro

Serra
2021

*Dedico este trabalho a todos que me apoiaram por traçar
este meu objetivo e especialmente minha mãe que sempre
me incentivou e apoiou nos estudos.*

AGRADECIMENTOS

Agradeço primeiramente a minha mãe que sempre me incentivou e me apoiou a estudar e que tem uma mente muito aberta em relação a educação e ensino. Agradeço também ao professor Dr. Fidelis Zanetti de Castro, pela dedicação, motivação e entusiasmo, sempre querendo fazer algo com vontade e da melhor qualidade possível. Agradeço à minha namorada Larissa Santos da Motta, que me ajudou bastante nas ideias para criação das figuras deste trabalho. Agradeço ao meu amigo Erick Ramos dos Santos, que é ex-aluno graduado em Engenharia de Controle de Automação no IFES e atualmente mestrando na área de IA, o qual me ajudou em dúvidas técnicas pontuais relativas ao aprendizado profundo e redes neurais. Agradeço também ao Robson por disponibilizar tempo no serviço em momentos primordiais para o desenvolvimento deste trabalho de conclusão de curso. Finalmente, agradeço a todos os amigos e professores por todo o suporte e contribuição em todo o curso.

Take your dreams seriously.

RESUMO

Neste trabalho de conclusão de curso, foi realizada uma análise investigativa relativa à determinação de similaridades entre pares de sentenças de renomados escritores da literatura norte-americana usando rede neurais siamesas com sub-redes *Long Short Term Memories* (LSTMs). As três bases de dados usadas para treinamento, validação e teste dos modelos foram construídas a partir da extração, limpeza e organização de 72600 sentenças de 35 obras literárias dos autores norte-americanos *William Cuthbert Faulkner*, *Ernest Miller Hemingway* e *Philip Milton Roth*. Antes de fornecer os pares de sentenças como entradas às redes neurais siamesas, foi realizado um processo de *word embedding* das palavras a partir de um modelo pré-treinado *Word2Vec*. O processo de extração de *features* das sentenças, bem como o de aprendizagem e a predição das similaridades foi realizado por meio de sub-redes LSTM usando a medida de similaridade de *Manhattan* acopladas em uma arquitetura siamesa de compartilhamento de pesos sinápticos. A escolha das LSTMs se deve ao fato de elas serem redes neurais recorrentes com memória de longo prazo comumente aplicadas com sucesso em tarefas da área de processamento da linguagem natural (PLN). Já a escolha da arquitetura siamesa deve-se à estratégia metodológica de comparação de sentenças par a par. Os resultados obtidos reforçam a dificuldade inerente à captura e identificação de um *ethos* literário usando representações densas não contextuais de palavras, mesmo usando arquiteturas baseadas em aprendizado profundo.

Palavras-chave: Processamento da Linguagem Natural, Redes neurais siamesas, Redes neurais LSTM, Medida de similaridade de *Manhattan*, Literatura norte-americana.

ABSTRACT

In this undergraduate thesis, an investigative analysis concerning the determination of similarities between sentence pairs of renowned writers of North American literature was carried out using siamese neural networks with *Long Short Term Memories* (LSTMs) subnets. The three databases used for training, validating, and testing the models were built by extracting, cleaning and organizing 72600 sentences from 35 literary works by American authors: *William Cuthbert Faulkner*, *Ernest Miller Hemingway* and *Philip Milton Roth*. Before providing the pairs of sentences as inputs to the siamese neural networks, a process of *word embedding* of words was carried out using a pre-trained model *Word2Vec*. The process of extracting *features* from the sentences, as well as learning and predicting similarities was performed through LSTM subnets using the *Manhattan* similarity measure coupled in a siamese architecture, that is, with sharing of synaptic weights. The adoption of LSTMs is due to the fact that they are recurrent neural networks with long-term memory commonly applied successfully in tasks of natural language processing (PLN) area. In turn, the choice of the siamese architecture is due to the methodological approach of pairwise sentences comparison. The results obtained reinforce the inherent difficulty in capturing and identifying a literary *ethos* using dense non-contextual representations of words, even using architectures based on deep learning.

Keywords: Natural Language Processing, Siamese Neural Networks, LSTM Neural Networks, *Manhattan* Similarity Measure, North American Literature.

LISTA DE FIGURAS

Figura 1 – Algumas aplicações do PLN.	13
Figura 2 – Crescimento do mercado do PLN por região continental em bilhões de dólares.	14
Figura 3 – Estágios de desenvolvimento do PLN antes do aprendizado profundo. .	20
Figura 4 – Estágios de desenvolvimento do PLN após a instauração do aprendizado profundo.	21
Figura 5 – Formas e categorizações da similaridade de textos.	22
Figura 6 – Representação geométrica da distância de <i>Manhattan</i> em um espaço bidimensional.	25
Figura 7 – Neurônio fundamental de uma RNA.	27
Figura 8 – Estrutura de uma RNA simples.	28
Figura 9 – Tipos de RNAs quanto ao fluxo de dados.	28
Figura 10 – Processo de aprendizado supervisionado de uma RNA.	29
Figura 11 – Processo de aprendizado não-supervisionado de uma RNA.	29
Figura 12 – Breve histórico das RNAs.	30
Figura 13 – Aplicações das RNAs por áreas.	31
Figura 14 – Interesse pelo termo <i>Deep Learning</i>	32
Figura 15 – Performance dos algoritmos de aprendizado profundo em relação à quantidade de dados.	33
Figura 16 – Estruturas de RNAs rasa e profunda.	33
Figura 17 – Arquitetura da primeira rede neural profunda conhecida treinada por Alexey Grigorevich Ivakhnenko.	34
Figura 18 – Rede neural recorrente básica “desdobrada”.	35
Figura 19 – Estruturas das células de uma RNN clássica e de uma LSTM.	37
Figura 20 – Célula de estado da LSTM.	38
Figura 21 – Portão de esquecimento da LSTM.	38
Figura 22 – Portão de entrada da LSTM.	38
Figura 23 – Portão de saída da LSTM.	39
Figura 24 – Arquitetura padrão de uma rede neural siamesa (RNS).	41
Figura 25 – Fluxo de funcionamento principal da rede siamesa utilizada.	43
Figura 26 – Autores romancistas da literatura norte-americana.	44
Figura 27 – Fluxograma do processo de filtragem e seleção das sentenças de textos extraídas das obras literárias.	49
Figura 28 – Exemplo de pré-processamento de uma sentença válida extraída da obra <i>A Fable</i> (1954), do autor Faulkner, seguindo as etapas do fluxograma da Figura 27.	53
Figura 29 – Esquema da estruturação das sentenças dos conjuntos de dados de treinamento e validação.	55

Figura 30 – Esquema da estruturação das sentenças dos conjuntos de dados de teste.	57
Figura 31 – Processo de operação do Word2Vec.	59
Figura 32 – Exemplo de um processo de <i>word embedding</i> de uma sentença de texto de um dos conjuntos de dados.	60
Figura 33 – Histogramas da quantidade de sentenças em função da quantidade de palavras de cada sentença.	63
Figura 34 – Arquitetura do modelo da rede neural siamesa.	64
Figura 35 – Comparação de frases retiradas do desafio <i>Kaggle</i> com as deste TCC.	69
Figura 36 – Partição das bases de dados utilizadas nas fases de treinamento, valida- ção e testes.	70
Figura 37 – Função principal de execução dos experimentos computacionais do módulo desenvolvido da aplicação chamado de <i>experiments.py</i> .	71
Figura 38 – Gráficos de acurácia e <i>loss</i> do treinamento e validação do <i>conjunto de dados D₁</i> .	72
Figura 39 – Gráficos de acurácia e <i>loss</i> do treinamento e validação do conjunto de dados <i>D₂</i> .	74
Figura 40 – Gráficos de acurácia e <i>loss</i> do treinamento e validação no conjunto de dados <i>D₃</i> .	77

LISTA DE TABELAS

Tabela 1 – Quantidade de sentenças por obra selecionada de Faulkner para a composição das nossas bases de dados.	45
Tabela 2 – Quantidade de sentenças por obras selecionadas de Hemingway para a composição das nossas bases de dados.	46
Tabela 3 – Quantidade de sentenças por obras selecionadas de Roth para a composição das nossas bases de dados.	47
Tabela 4 – Quantidade total de sentenças extraídas por autor para o processo de treinamento, validação e teste da rede neural.	47
Tabela 5 – Exemplos de palavras após a aplicação das técnicas de lematização e <i>stemming</i> , respectivamente.	51
Tabela 6 – Frases com stopwords × Frases sem stopwords.	52
Tabela 7 – Exemplos de sentenças do arquivo CSV do conjunto de dados D_3 de treinamento e validação.	56
Tabela 8 – Exemplos de sentenças retiradas do arquivo CSV do conjunto de dados D_3 (sem stopwords e lematizado) de teste.	58
Tabela 9 – Tamanhos máximos das sequências escolhidas para cada conjunto de dados.	62
Tabela 10 – Hiperparâmetros do modelo de aprendizagem testados na ferramenta <i>Hyperas</i>	66
Tabela 11 – Configurações dos computadores utilizados para execução dos experimentos.	67
Tabela 12 – Tabela de similaridades médias entre combinações de sentenças de autores da base de dados D_1 usando tamanho da sequência 17.	73
Tabela 13 – Tabela de similaridades médias entre combinações de frases de autores da base de dados D_1 usando tamanho da sequência 14.	73
Tabela 14 – Tabela de similaridades médias entre combinações de frases de autores da base de dados D_1 usando tamanho da sequência 10.	73
Tabela 15 – Tabela de similaridades médias entre combinações de frases de autores do conjunto de dados D_2 usando tamanho da sequência 9.	75
Tabela 16 – Tabela de similaridades médias entre combinações de frases de autores do conjunto de dados D_2 usando tamanho da sequência 7.	75
Tabela 17 – Tabela de similaridades médias entre combinações de frases de autores do conjunto de dados D_2 usando tamanho da sequência 5.	75
Tabela 18 – Tabela de similaridades médias entre combinações de frases de autores no conjunto de dados D_3 usando tamanho da sequência 9.	77
Tabela 19 – Tabela de similaridades médias entre combinações de frases de autores no conjunto de dados D_3 usando tamanho da sequência 7.	78

Tabela 20 – Tabela de similaridades médias entre combinações de frases de autores
no conjunto de dados D_3 usando tamanho da sequência 5. 78

LISTA DE SIGLAS

ALPAC	– Automatic Language Processing Advisory Committee
CBOW	– Continous Bag of Words
CNN	– Convulational Neural Network
CSV	– Comma-Separated Values
EUA	– Estados Unidos da América
GRU	– Gated Recurrent Units
HTRS-Rank	– Hot Topics Rising Star Rank
IEEE	– Instituto de Engenheiros Eletricistas e Eletrônicos
Ifes	– Instituo Federal de Educação, Ciênciа e Tecnologia do Espírito Santo
LCS	– Longest Common Sequence
LDA	– Latent Dirichlet Allocation
LSA	– Latent Semantic Similarity
LSTM	– Long Short-Term Memory
MaLSTM	– Manhattan Long Short-Term Memory
NCD	– Normalized Compression Distance
NGD	– Normalized Google Distance
NLTK	– Natural Language ToolKit
PLN	– Processamento da Linguagem Natural
RNA	– Rede Neural Artificial
RNN	– Recurrent Neural Network
RNS	– Rede Neural Siamesa
SELU	– Scaled Exponential Linear Unit
STS	– Semântica de Similaridade Textual
TCC	– Trabalho de Conclusão de Curso
TF-IDF	– Term Frequency - Inverse Term Frequency

TF-IDF – Term Frequency - Inverse Line Frequency

Web – World Wide Web

SUMÁRIO

1	INTRODUÇÃO	13
1.1	OBJETIVOS	16
1.1.1	Objetivo Geral	16
1.1.2	Objetivos Específicos	16
1.2	METODOLOGIA DO TRABALHO	17
1.3	ESTRUTURA DO TRABALHO	18
2	REFERENCIAL TEÓRICO	19
2.1	PROCESSAMENTO DA LINGUAGEM NATURAL	19
2.2	SIMILARIDADE DE TEXTOS	22
2.2.1	Distâncias, Similaridades e Manhattan	24
2.3	REDES NEURAIS	26
2.3.1	Redes Neurais Profundas	32
2.3.1.1	Redes Neurais <i>Long Short-Term Memory</i>	35
2.4	REDES NEURAIS SIAMESAS	40
3	MATERIAIS E MÉTODOS	43
3.1	ESTRUTURA GERAL DOS EXPERIMENTOS COMPUTACIONAIS . .	43
3.2	BASE DE DADOS	44
3.2.1	William C. Faulkner	45
3.2.2	Ernest M. Hemingway	45
3.2.3	Philip M. Roth	46
3.3	PRÉ-PROCESSAMENTO TEXTUAL	48
3.3.1	Técnicas de pré-processamento aplicadas	51
3.4	ESTRUTURAÇÃO DO CONJUNTO DE DADOS	54
3.4.1	Word Embedding Utilizado	58
3.5	DEFINIÇÃO DE PARÂMETROS E HIPERPARÂMETROS	61
3.5.1	Definição do Tamanho Máximo de Sequência	61
3.5.2	Arquitetura da Rede Siamesa	63
3.5.3	Hiperparâmetros definidos usando uma heurística Bayesiana . .	65
3.6	RECURSOS COMPUTACIONAIS UTILIZADOS	66
3.7	SOBRE A REPRODUTIBILIDADE DESTE TRABALHO	68
4	EXPERIMENTOS, RESULTADOS E DISCUSSÃO	69
4.1	RESULTADOS OBTIDOS COM O CONJUNTO DE DADOS D_1	71
4.2	RESULTADOS OBTIDOS COM O CONJUNTO DE DADOS D_2	74
4.3	RESULTADOS OBTIDOS COM O CONJUNTO DE DADOS D_3	76
4.4	ANÁLISES FINAIS DOS RESULTADOS	78
5	CONSIDERAÇÕES FINAIS	81
5.1	TRABALHOS FUTUROS	82
	REFERÊNCIAS	83

1 INTRODUÇÃO

Atualmente é comum a utilização de ferramentas tecnológicas que possuem a capacidade de “interpretar” (em algum sentido) a linguagem natural humana, tais como *chatbots*, tradutores de documentos, corretores textuais em *smartphones* ou na *Web*, editores de textos e mecanismos de buscas por assunto de interesse em páginas *Web*. Esta variedade de ferramentas que utilizamos no nosso cotidiano só foi possível ser concebida graças aos avanços científicos e tecnológicos providos pela subárea de pesquisa da Inteligência Computacional denominada *Processamento da Linguagem Natural* (PLN).

Segundo a cientista Elizabeth DuRoss Liddy, professora emérita de Ciência da Informação da Escola de Estudos da Informação da Universidade de Siracusa, na Itália, o Processamento da Linguagem Natural consiste em um conjunto de “teorias motivadas por uma série de técnicas computacionais para análise e representação de textos decorrentes da linguagem natural. Essas técnicas são utilizadas com o objetivo de processar linguagens humanas para diversas aplicações” (LIDDY, 2001). Nesse sentido, o PLN possibilita que computadores manipulem, analisem e interpretem a linguagem natural humana, criando condições que permitem uma comunicação entre máquina e ser humano.

Figura 1 – Algumas aplicações do PLN.



Fonte: Elaborada pelo autor (2020).

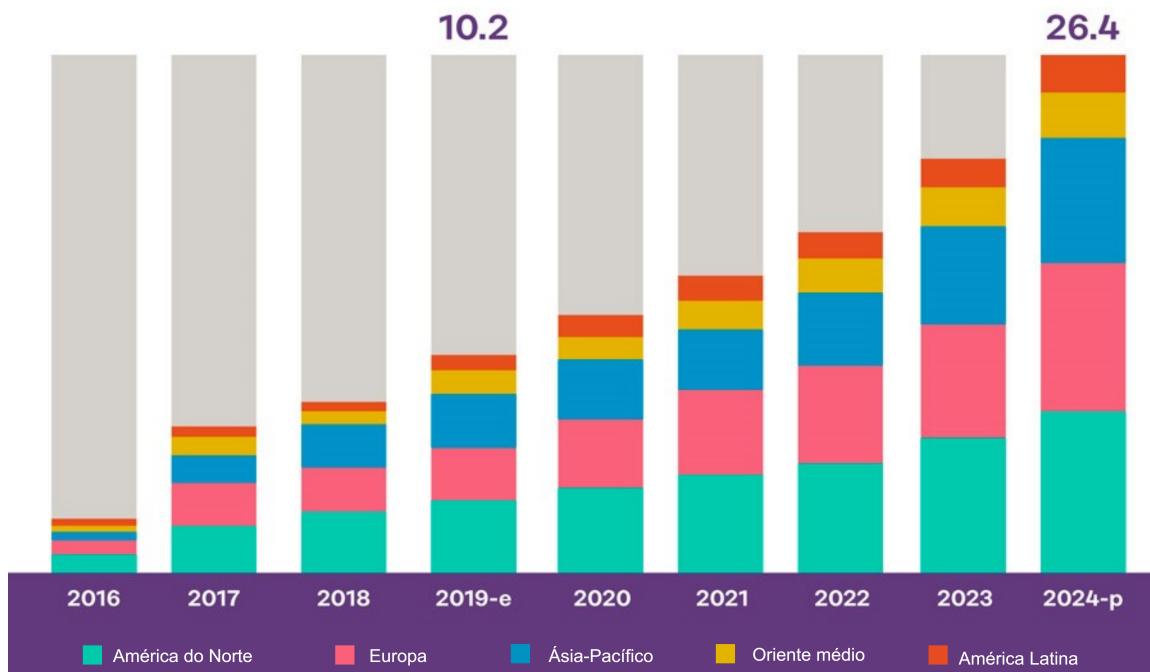
Ao longo dos anos, a geração e a disponibilização de dados não estruturados cresceu significativamente. Para se ter uma ideia, cerca de 80% dos dados gerados por empresas são não estruturados, os quais se tornam disponíveis principalmente em mídias sociais ou são obtidos a partir de conversas com representantes de atendimento (BORGES, 2018).

As corporações estão recorrendo ao PLN no intuito de obterem ferramentas para capturar

e organizar informações textuais e, a partir delas, extrair informações para tomadas de decisão inteligentes em diversos cenário visando aumentar a sua competitividade no mercado.

A Figura 2 ilustra o crescimento do mercado na área de PLN por regiões continentais entre os anos de 2016 e 2020 e uma previsão de crescimento até o final de 2024.

Figura 2 – Crescimento do mercado do PLN por região continental em bilhões de dólares.



Fonte: Adaptada de (JASSOVA, 2020).

Nota-se que regiões com maior concentração de países desenvolvidos possuem uma taxa de crescimento em PLN maior do que regiões com menor concentração. Isso se dá, ao nosso ver, pelo fato de os países desenvolvidos alocarem uma maior quantidade de recursos de investimento para áreas tecnológicas e científicas, em comparação aos países em desenvolvimento.

A similaridade de textos (*text similarity*) é uma área de estudo em PLN cujo objetivo é fornecer uma medida de “proximidade” entre textos no âmbito do desenvolvimento de uma aplicação específica como, por exemplo, rastreamento de tópicos, classificação e sumarização de textos, agrupamento de documentos e detecção de plágio (SIEG, 2018). Uma das principais aplicações de medidas de similaridade em PLN se dá na área de detecção de plágio. Aplicações voltadas para detecção de plágio são fundamentais para estimar/determinar a originalidade e a autoria de textos, assegurando a integridade e a propriedade intelectual produzida pelo autor que o escreveu (BARRÓN-CEDEÑO; GUPTA; ROSSO, 2013; MEMON, 2020).

Neste Trabalho de Conclusão de Curso (TCC) utilizamos técnicas de PLN e de aprendizado de máquina para analisar sentenças extraídas de obras literárias escritas em língua inglesa e determinar a similaridade entre eles (par a par) no contexto da nossa base de dados.

BEKMIRZAEV et al. (2017) utilizaram as medidas de similaridade do cosseno e de Jaccard com intuito de analisar poemas chineses clássicos do século XV. Para isso, foram selecionados nove livros de poemas analisando-se a associação de autor com poema. BEKMIRZAEV e seus colaboradores utilizaram as transformações clássicas *Term frequency - Inverse term frequency* (TF-IDF) e *Term frequency - Inverse line frequency* (TF-ILF) para converter os poemas para um formato numérico. Seus resultados apresentaram similaridades variando de 0,0919, usando a similaridade *Jaccard* com a abordagem TF-IDF, a 0,2794, utilizando a similaridade do cosseno com a abordagem TF-ILF (BEKMIRZAEV; KIM; LEE, 2017).

RAHGOZAR e INKPEN (2019), também no contexto de análise de textos literários, utilizaram uma abordagem semântica e de agrupamento homotético baseada em uma variação do algoritmo *K-means* com a finalidade de automatizar um processo de rotulação dos poemas do poeta lírico persa Hafez (Khāja Šamsu d-Dīn Muammad Ḥāfez-e Šīrāzī), tarefa que era usualmente feita à mão livre. Os autores compararam a performance do seu modelo utilizando diferentes estratégias de extração de *features*, tais como: a abordagem semântica *Doc2Vec*, a técnica *Latent Dirichlet Allocation* (LDA) e o algoritmo clássico TF-IDF, bem como algumas de suas combinações. A melhor performance foi obtida usando-se a estratégia *Doc2Vec*, aproximadamente 9 vezes maior do que usando a técnica LDA, e aproximadamente 15 vezes maior do que com o algoritmo TF-IDF, tomando como base a métrica de avaliação *Silhouette*, utilizada para interpretar e validar a consistência de dados em *clusters*. Mesmo diante do resultado evidentemente superior que o *Doc2Vec* proporcionou, o valor da métrica *Silhouette*, numa escala de -1 a 1, não ultrapassou 0,57, o que ilustra o quanto difícil é o problema do ponto de vista do PLN (RAHGOZAR; INKPEN, 2019).

SHANG et. al (2019) realizaram um estudo comparativo entre a obra do poeta irlandês William Butler Yeats (1865-1939) e os poetas românticos ingleses William Blake (1757 - 1827), William Wordsworth (1770 - 1850), John Keats (1795 - 1821) e Lord Byron (1788 - 1824). Nesse estudo, os autores propuseram um modelo para determinar a “influência” de um autor sobre o outro nos quesitos estilo e gênero poético. Especificamente, um modelo foi construído para comparar a obra de Yeats com as dos poetas ingleses explorando suas semelhanças em três aspectos: intertextualidade, elementos formais - incluindo rima e métrica -, e sentimento. A base de dados que alimentou a pesquisa incluiu 130 obras de Yeats de três coleções, 216 obras de Blake, 119 obras de Byron, 53 obras de Keats e 469 obras de Wordsworth. As métricas para mensurar a intertextualidade entre pares de

frases foram definidas pelos próprios proponentes da pesquisa: um *score* logarítmico e uma medida denominada *verse-pair value* (SHANG; ZHANG; HUANG, 2019). Para analisar os sentimentos presentes nos pares de frases foi utilizada a biblioteca *TextBlob*, do Python (LORIA, 2018).

ZHANG et al. utilizaram um método de agrupamento hierárquico com o objetivo de descobrir diferenças e semelhanças entre os estilos literários dos poetas Thomas Hardy (1840 - 1928), Oscar Wilde (1854 - 1900), Robert Browning (1812 - 1889), William Butler Yeats (1865 - 1939) e Rabindranath Tagore (1861 - 1941). Para extração de *features*, limitaram-se às 20 palavras mais frequentes nas obras desses autores e criaram vetores de representação em dimensão 80, 100 e 120, todos baseados na técnica *Word2Vec*. O modelo proposto por eles obteve dois *clusters* principais: um formado por Hardy, Wilde e Browning; outro pelos autores remanescentes. Os proponentes do trabalho afirmam que os resultados numéricos refletem o fato de que os estilos dos poetas são, de certo modo, semelhantes sob uma perspectiva literária ao afirmarem que os autores têm visões semelhantes acerca da interpretação da tragédia e dos conflitos entre homens e a sociedade (ZHANG; GAO et al., 2017).

Nossa revisão de literatura verificou que são raros os trabalhos que tentam capturar o *ethos*¹ artístico-literário autoral por meio de técnicas que envolvem o uso de redes neurais profundas e extração de *features* textuais semânticas e suas representações em vetores densos, como é o caso da técnica *Word2Vec*, por exemplo. São ainda mais raros trabalhos que associam o que foi mencionado acima ao uso de medidas de similaridade.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Analisar a performance de uma rede neural siamesa LSTM para obtenção de similaridades de *Manhattan* entre pares de sentenças de textos literários de escritores da literatura norte-americana.

1.1.2 Objetivos Específicos

Os objetivos específicos elencados para se atingir o objetivo geral proposto são:

- Criar, organizar e limpar uma base de dados composta por textos literários de três escritores da literatura norte-americana;

¹ Na *Análise do Discurso*, a palavra *ethos* é utilizada para representar a característica de continuidade tênue presente nas obras de um autor ao longo do tempo. No dicionário *Houaiss*, a palavra *ethos* se refere ao “modo de ser, natureza (emocional, moral, intelectual) habitual de um indivíduo.”

- Derivar, a partir da base de dados criada, um segundo conjunto de dados aplicando remoção de *stopwords*;
- Derivar um terceiro conjunto de dados aplicando remoção de *stopwords* e lematização;
- Representar as palavras dos três conjuntos de dados por vetores multidimensionais usando a técnica de *word embedding Word2Vec*;
- Implementar uma rede neural siamesa LSTM voltada para a determinação da similaridade de *Manhattan* entre pares de sentenças literárias de combinações de autores;
- Realizar análises e discussões sobre a viabilidade da solução proposta levando-se em conta diferentes tipos de ajustes hiperparamétricos no modelo utilizado;
- Disponibilizar publicamente as bases de dados organizadas por meio do repositório público de dados *Mendeley Data*;
- Disponibilizar publicamente os algoritmos utilizados neste trabalho por meio da plataforma de gerenciamento de projetos *Github*.

1.2 METODOLOGIA DO TRABALHO

Nossa pesquisa caracteriza-se como teórico-aplicada, pois estudamos conceitos e técnicas que envolvem o uso de redes neurais siamesas LSTM no contexto do PLN e, além disso, tomando-os como base, realizamos experimentos computacionais para determinar as similaridades entre pares de textos de autores.

A análise de dados é quantitativa/qualitativa, uma vez que os experimentos computacionais fornecem resultados numéricos correspondentes a similaridades de *Manhattan*. os quais são interpretados numericamente e analisados descritivamente em diálogo dos resultados obtidos com os objetivos traçados e conjecturas enunciadas.

A linguagem de programação escolhida para o desenvolvimento dos experimentos computacionais deste trabalho foi o *Python*, por ser uma linguagem de alto nível composta por uma ampla comunidade e possuir bibliotecas para suporte ao PLN, dentre elas, a NLTK (*Natural Language ToolKit*). As bases de dados utilizadas para treinamento, validação e teste foram construídas a partir de trechos de obras literárias disponibilizadas na *Internet*.

O percurso metodológico que originou este trabalho é composto pelas seguintes atividades:

- Revisão de literatura sobre a rede neural siamesa LSTM;
- Revisão de literatura sobre técnicas de *word embedding*;

- Criação, organização e limpeza de três bases de dados compostas por textos literários de William Cuthbert Faulkner, Ernest Miller Hemingway e Philip Milton Roth;
- Implementação de uma rede neural siamesa LSTM voltada para a determinação da similaridade de *Manhattan* entre pares de sentenças de combinações desses autores;
- Realização de ajustes hiperparamétricos no modelo implementado utilizando um algoritmo baseado em otimização Bayesiana;
- Análise do desempenho da rede neural proposta na tarefa de determinação da similaridade de *Manhattan* entre pares de sentenças de combinações dos autores selecionados.

1.3 ESTRUTURA DO TRABALHO

O restante deste trabalho está dividido em quatro capítulos. O Capítulo 2 contém a fundamentação teórica. Nele, são apresentados conceitos relativos ao processamento da linguagem natural, similaridade de textos, incluindo a medida de similaridade de *Manhattan*, e uma revisão dos modelos de redes neurais usadas para realização dos experimentos computacionais. O Capítulo 3 fornece os detalhes metodológicos da pesquisa, incluindo uma breve revisão bibliográfica sobre os autores estudados, detalhes sobre a criação dos conjuntos de dados, seu pré-processamento e vetorização de palavras usando a técnica *Word2Vec*, a estrutura geral dos experimentos computacionais elaborados, a definição de parâmetros e hiperparâmetros do modelo, os recursos computacionais utilizados e alguns detalhes acerca da implementação. No Capítulo 4, são realizadas as análises dos resultados obtidos nos experimentos computacionais. No último capítulo, são tecidas as considerações finais e elencadas as possibilidades de trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo são apresentadas as fundamentações dos conceitos que são utilizados como base ao longo de todo o trabalho. Primeiramente, é abordada a importância da área de PLN para a sociedade. Em seguida, apresentamos um histórico do PLN atrelado ao desenvolvimento dos modelos de aprendizado de máquina a ele correlacionados. Dando sequência, tratamos do conceito de similaridades textuais e da medida de similaridade de *Manhattan*. Por fim, são apresentados detalhes sobre as redes neurais artificiais (RNAs), incluindo suas características e arquiteturas, com ênfase na rede neural siamesa (RNS) LSTM, que é aplicada no trabalho.

2.1 PROCESSAMENTO DA LINGUAGEM NATURAL

O Processamento da Linguagem Natural (PLN) está presente em várias áreas do nosso cotidiano relacionando-se com diversos aspectos da comunicação humana de maneira computacional. GONZALEZ e LIMA (2003) afirmam que o PLN tem como objetivo aproximar o computador da linguagem natural humana ao propiciar uma espécie de comunicação homem-máquina (GONZALEZ; LIMA, 2003).

As pesquisas na área do PLN se dão desde o final da década de 1940 e, neste TCC, são divididas em dois principais períodos: antes do instauração do paradigma de aprendizado profundo e durante o seu desenvolvimento (LOUIS, 2020).

Em 1949, foi escrito por Warren Weaver o memorando que trouxe a primeira aparição de uma aplicação computacional baseada em linguagem natural, a tradução automática. Seu memorando foi uma das publicações mais relevantes para a área na época (HUTCHINS, 2000). Inspirando diversos outros projetos em tradução automática, por volta de 1955 o experimento de Georgetown permitiu a tradução automática de mais de 60 frases da língua russa para a inglesa (HUTCHINS; DOSTERT; GARVIN, 1955).

Apesar do sucesso obtido, pesquisadores notaram que as máquinas de tradução automática não levavam em consideração a ambiguidade lexical e semântica da linguagem natural, além de que gerava resultados não totalmente correspondentes ao que se esperava. Desse modo, o problema de tradução automática foi admitido como muito mais complexo e surgiu a necessidade de uma nova estratégia de abordagem. Foi então que em 1957, Chomsky, ao introduzir a ideia da gramática generativa, aproximou a linguística convencional da tradução automática, baseando-se em regras de estruturas sintáticas (CHOMSKY, 1957).

Mesmo com o desenvolvimento de técnicas de análise de algoritmos e teoria sintática da linguagem, por volta de 1966 o *Automatic Language Processing Advisory Committee* (ALPAC) divulgou um relatório que concluiu que os altos investimentos em pesquisas na área de tradução automática não correspondiam às necessidades almejadas e aos benefícios

esperados. A partir da publicação desse relatório, investimentos na área de tradução automática sofreram uma diminuição drástica que também impactou negativamente o desenvolvimento de pesquisas na área do PLN (PIERCE et al., 1966).

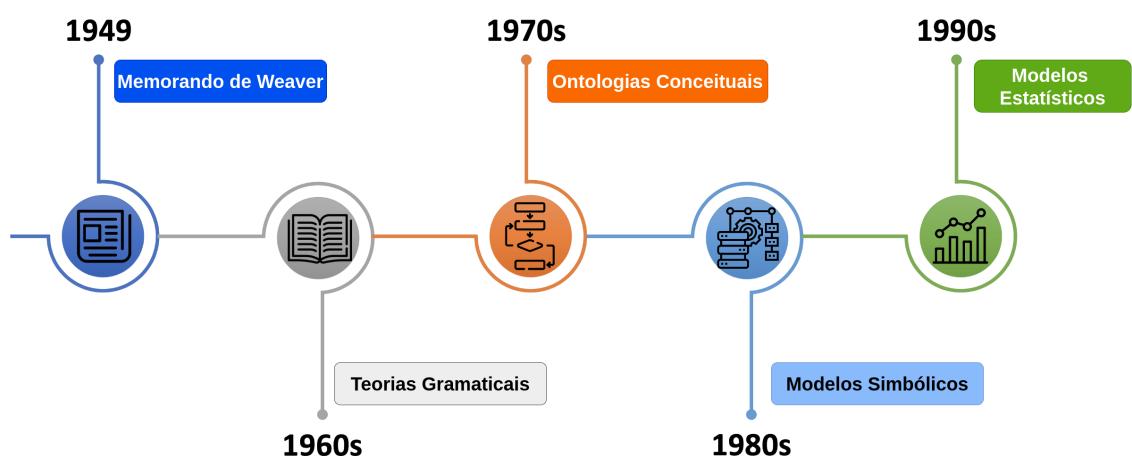
Todavia, durante o final da década de 1960 e início da década de 1970, trabalhos voltados para representação semântica da linguagem se tornaram destaques. Alguns deles forneciam representações semânticas e elaboravam teorias para explicar anomalias sintáticas, como no caso de *Case Grammar* (FILLMORE; FILLMORE, 1968), *Transition Network Grammars for Natural Language Analysis* (WOODS, 1970), *Experiments on Semantic Memory and Language Comprehension* (COLLINS; QUILLIAN, 1972) e *Conceptual dependency: A theory of Natural Language Understanding* (SCHANK, 1972).

Durante a segunda metade da década de 1970 surgiram novas ideias que buscavam estruturar informações de forma a facilitar sua interpretação por parte do computador, as chamadas ontologias conceituais. Um exemplo desta abordagem está presente no artigo *POLITICS: Automated ideological reasoning* (JR, 1978).

Na década de 1980 foram utilizados modelos que envolvem abordagens simbólicas, nos quais regras e gramáticas complexas eram utilizadas para analisar fenômenos linguísticos, como abordam os artigos *Passing markers: A theory of contextual influence in language comprehension* (CHARNIAK, 1983) e *TEAM: An experiment in the design of transportable natural-language interfaces* (GROSZ et al., 1987).

a década de 1990, houve um grande destaque nos modelos estatísticos, capazes de tomar decisões baseadas no aprendizado estatístico, fornecendo resultados promissores. Obras que se destacam nessa década são: *Statistical parsing of messages* (CHITRAO; GRISHMAN, 1990) e *Word-sense disambiguation using statistical methods* (BROWN et al., 1991). Esse tipo de abordagem atraiu diversos pesquisadores e ergueu a porta de entrada para a era do PLN baseado no aprendizado profundo.

Figura 3 – Estágios de desenvolvimento do PLN antes do aprendizado profundo.



Fonte: Adaptada de (LOUIS, 2020).

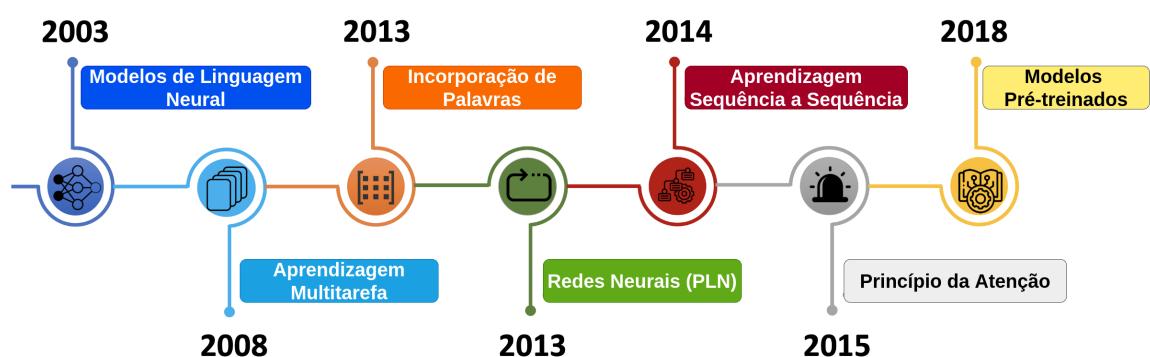
Em 2003, foi proposto o primeiro modelo de rede neural artificial *feedforward* voltada para o PLN. Nesse mesmo ano foi introduzido uma estratégia para transformação de palavras em vetores multidimensionais, comumente conhecida como *word embedding* (BENGIO et al., 2003). Em 2008, foi introduzido o aprendizado baseado no método multitarefa, a partir do qual é possível a resolução de várias tarefas ao mesmo tempo usando uma dada sentença (COLLOBERT; WESTON, 2008).

O ano de 2013 foi marcado por duas contribuições importantes para o PLN. A primeira foi a introdução de um dos mais famosos modelos de incorporação de palavras chamado Word2Vec, que trouxe mais velocidade e precisão, do que modelo proposto por BENGIO e colaboradores em 2003 (MIKOLOV et al., 2013). A segunda foi a popularização de dois tipos de redes neurais profundas voltadas para o PLN: as convolucionais (CNNs) e as recorrentes (RNNs).

Em 2014 foi proposta uma metodologia de aprendizado de máquina que se baseia em mapear, de ponta a ponta, uma sequência para outra sequência utilizando uma rede neural, a qual é chamada aprendizado sequência a sequência (*seq2seq*) (SUTSKEVER; VINYALS; LE, 2014). Essa abordagem foi tão bem sucedida que poucos anos depois a *Google* decidiu introduzi-la em seu tradutor automático, o *Google Translate*. No ano seguinte, em 2015, foi concebido o princípio da atenção, o qual melhorou a eficiência do aprendizado (*seq2seq*) compactando todo conteúdo de origem em um vetor de tamanho fixo (BAHDANAU; CHO; BENGIO, 2014). Este efeito foi um dos principais pontos que tornou a tradução automática neural superior aos modelos clássicos de tradução automática.

O ano de 2018 foi marcado pela adoção dos modelos de linguagem pré-treinados baseados em contexto. Alguns exemplos de artigos que se remetem a esses modelos são: *Bert: Pre-training of deep bidirectional transformers for language understanding* (DEVLIN et al., 2018), *Universal language model fine-tuning for text classification* (HOWARD; RUDER, 2018) e *Language models are unsupervised multitask learners* (RADFORD et al., 2019).

Figura 4 – Estágios de desenvolvimento do PLN após a instauração do aprendizado profundo.



Fonte: Adaptada de (LOUIS, 2020).

Com base na revisão histórica do PLN apresentada, pode-se notar a grande relevância que essa área no desenvolvimento de pesquisas científicas e no desenvolvimento de aplicações tecnológicas ao longo dos últimos anos. Nota-se, atualmente, que as empresas que lidam com grandes volumes de dados tendem a utilizar ferramentas do PLN para analisar informações para tomadas de decisões estratégicas (WOLFF, 2020).

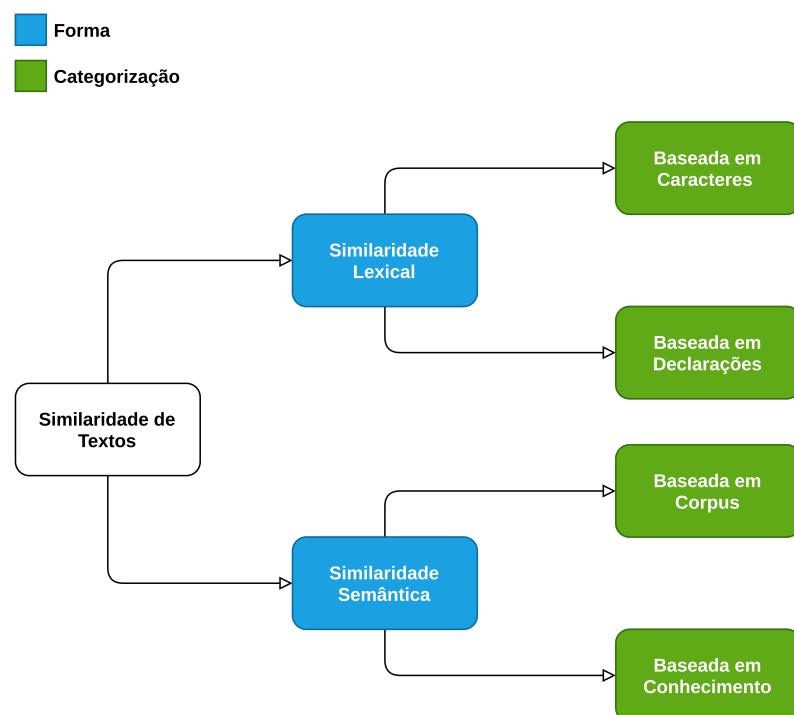
Dentre as várias aplicações que utilizam o PLN, destacamos a classificação de textos (MCCALLUM; NIGAM et al., 1998; JOULIN et al., 2016), a recuperação de informações (BAEZA-YATES; RIBEIRO-NETO et al., 1999; CROFT; METZLER; STROHMAN, 2010), a sumarização de textos (BARZILAY; ELHADAD, 1999; MAYBURY, 1999) e a identificação de autoria (ZHENG et al., 2006; HOUVARDAS; STAMATATOS, 2006).

2.2 SIMILARIDADE DE TEXTOS

A similaridade de textos é um processo pelo qual é determinada uma relação de proximidade/afastamento entre duas ou mais instâncias de textos (PRADHAN; GYANCHANDANI; WADHVANI, 2015). Ela desempenha um papel fundamental na categorização de textos literários e documentos, na classificação de textos e na análise de sentimentos.

A similaridade de textos pode ser dividida em dois tipos. A lexical, em que instâncias de textos são consideradas semelhantes caso contenham sequências de caracteres próximas, e a semântica, na qual instâncias de textos são consideradas semelhantes baseando-se em seus significados e em contextos mais gerais (GOMAA; FAHMY et al., 2013).

Figura 5 – Formas e categorizações da similaridade de textos.



Fonte: Elaborada pelo autor (2021).

Como ilustrado na Figura 6, a similaridade lexical é categorizada em duas formas: baseada em caracteres e baseada em declarações. Ambas possuem como característica fundamental o cálculo da de uma medida de similaridade através da correspondência ou comparação aproximada de composição de caracteres ou sequências de *strings*. Exemplos de medidas baseadas em similaridade lexical incluem *Longest Common Sequence (LCS)*, o *N-gram*, similaridade do cosseno, similaridade de Jaccard e similaridade de Manhattan (GOMAA; FAHMY et al., 2013).

A similaridade semântica, por sua vez, pode ser também apresentada de duas formas: baseada em *corpus* e baseada em conhecimento. A baseada em *corpus* é caracterizada por estabelecer a similaridade entre palavras de uma grande coleção de escritos de diferentes domínios de conhecimento. Já a baseada em conhecimento define a similaridade através das relações entre as palavras e por informações provenientes de redes semânticas, tal como a *Word Net*. Medidas que as compõem são: *Latent Semantic Similarity (LSA)*, *Normalized Google Distance (NGD)* e *Normalized Compression Distance (NCD)* (PRADHAN; GYANCHANDANI; WADHVANI, 2015).

No ano de 2004 foi publicado o artigo intitulado *MEAD - a platform for multidocument multilingual text summarization*, que introduz um ambiente de sumarização de textos capaz de realizar resumos de vários documentos em diversas linguagens (RADEV et al., 2004). Na época era utilizado em páginas *Web* para detecção de novidades e retorno de buscas, e foi baixado por mais de 500 organizações.

No artigo intitulado *Recurrent Convolutional Neural Networks for Text Classification*, foi proposto um sistema de classificação de textos baseado na extração de *features* sem recursos projetados por humanos (LAI et al., 2015). Os resultados obtidos mostraram que o método adotado é superior a diversos métodos da época, período em torno 2015, para um variado conjunto de dados.

Em 2021, no artigo intitulado *Finding rising stars through hot topics detection*, é proposto um modelo de detecção de tópicos capaz de encontrar possíveis acadêmicos juniores em ascensão usando o método *Hot Topics Rising Star Rank (HTRS-Rank)* (DAUD et al., 2021). Os resultados obtidos nos experimentos indicam que o método *HTRS-Rank* possui melhor desempenho em comparação aos seus concorrentes voltados para autores famosos e com alto número de citações.

Encontrar a similaridade entre frases, palavras, parágrafos e documentos é uma parte necessária em similaridade de textos (VIJAYMEENA; KAVITHA, 2016). Por este motivo, existem as *medidas de similaridade*, as quais permitem calcular níveis de “proximidade” entre textos. Na próxima seção será abordado o conceito de medida de similaridade em Matemática e apresentada a medida de similaridade que foi utilizada em nossa pesquisa:

a medida de *similaridade de Manhattan*.

2.2.1 Distâncias, Similaridades e Manhattan

Em primeiro lugar, é importante destacar, do ponto de vista matemático, a diferença entre medida de distância (métrica) e medida de similaridade. Uma medida de distância $M : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ é uma função que satisfaz as seguintes propriedades para todos A e B em \mathbb{R}^n :

1. $M(A, B) = 0 \Leftrightarrow A = B$.
2. $M(A, B) \geq 0$.
3. $M(A, B) = M(B, A)$.
4. $M(A + B) \leq M(A) + M(B)$.

Em palavras, 1. diz que a distância entre dois vetores é igual a zero se, e somente se, eles forem iguais. A propriedade 2. diz que não é possível o resultado de uma distância ser negativo. A propriedade 3. garante que a ordem dos elementos nos quais é aplicada a função de distância não altera o seu resultado. Por fim, a propriedade 4. é a conhecida desigualdade triangular.

É importante destacar que uma medida de distância pode gerar qualquer resultado não negativo, isto é, de zero ao infinito. Exemplos de medidas de distância incluem a distância Euclidiana, a distância de Manhattan, a distância do cosseno, a distância de Mahalanobis e a distância de Minkowski.

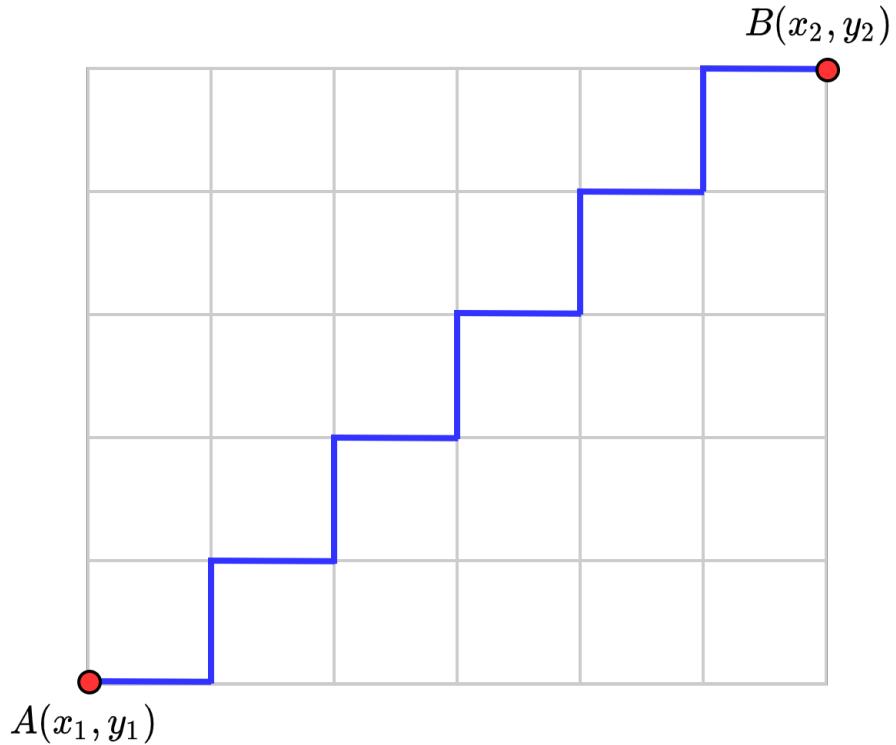
Como exemplo, apresentamos a distância de Manhattan. Sejam $V = (v_1, v_2, \dots, v_n)$ e $W = (w_1, w_2, \dots, w_n)$ vetores em \mathbb{R}^n . A distância de Manhattan (a qual é também conhecida por distância do quarteirão da cidade, métrica L1 ou métrica do táxi) entre V e W , denotada por $|V - W|_1$, é dada por

$$|V - W|_1 = \sum_{i=1}^n |v_i - w_i|.$$

Essa distância é caracterizada como a menor possível entre dois pontos em um hiperespaço em formato de grade (GOHRANI, 2019).

Para fins ilustrativos, a Figura 6 fornece uma representação da distância de Manhattan (em azul) em dois vetores A e B em um espaço bidimensional.

Figura 6 – Representação geométrica da distância de *Manhattan* em um espaço bidimensional.



Fonte: Adaptada de (ZORNOZA, 2020)

Por sua vez, em Estatística e campos relacionados, uma medida de similaridade é uma função de valor real que quantifica o quanto similares são dois objetos, os quais, no nosso contexto, são subconjuntos de números reais apresentados em formato de vetores n-dimensionais. Dizemos que $\mathcal{S} : \mathbb{R}^n \times \mathbb{R}^n \rightarrow [0, 1]$ é uma medida de similaridade, se para quaisquer vetores A e B em \mathbb{R}^n , $n > 0$, valem as seguintes propriedades:

- 1'. $\mathcal{S}(A, A) = 1$.
- 2'. $\mathcal{S}(A, B) = \mathcal{S}(B, A)$.

Em palavras, 1' diz que a similaridade entre dois vetores idênticos é igual a 1. A propriedade 2' garante que a ordem dos elementos nos quais é aplicada a função de similaridade não altera o seu resultado. Por fim, o conjunto imagem na definição de função de similaridade garante que $0 \leq \mathcal{S}(A, B) \leq 1$, quaisquer que sejam os vetores A e B em \mathbb{R}^n .

Diferentemente de uma medida de distância, uma medida de similaridade tem seus resultados limitados ao intervalo fechado $[0, 1]$. Quanto mais próximo o resultado da similaridade entre dois vetores for de 1, mais “próximos” em algum sentido são os vetores. Exemplos de medidas de similaridade incluem a similaridade do cosseno, a similaridade de

Jaccard e a similaridade baseada na distância de Manhattan, que denotaremos neste TCC simplesmente por similaridade de Manhattan, a qual foi a utilizada em nossos experimentos computacionais.

A similaridade de Manhattan entre dois vetores V e W em \mathbb{R}^n é dada pela expressão

$$S(V, W) = e^{-|V - W|_1},$$

em que $|V - W|_1$ é a distância de Manhattan entre V e W .

Uma variedade de algoritmos baseados em aprendizado de máquina, tais como *K-means clustering* (STEINHAUS et al., 1956) e *K-nearest neighbors* (SILVERMAN; JONES, 1989), podem utilizar medidas de similaridade ou de distância para mensurar as semelhanças entre duas ou mais instâncias de dados (BROWNLEE, 2020a). A escolha de uma medida de similaridade ou de distância adequada ao contexto de um problema de estudo pode melhorar a performance do modelo de aprendizado, desempenhando, portanto, um papel fundamental nos algoritmos de aprendizado de máquina (SHARMA, 2020a).

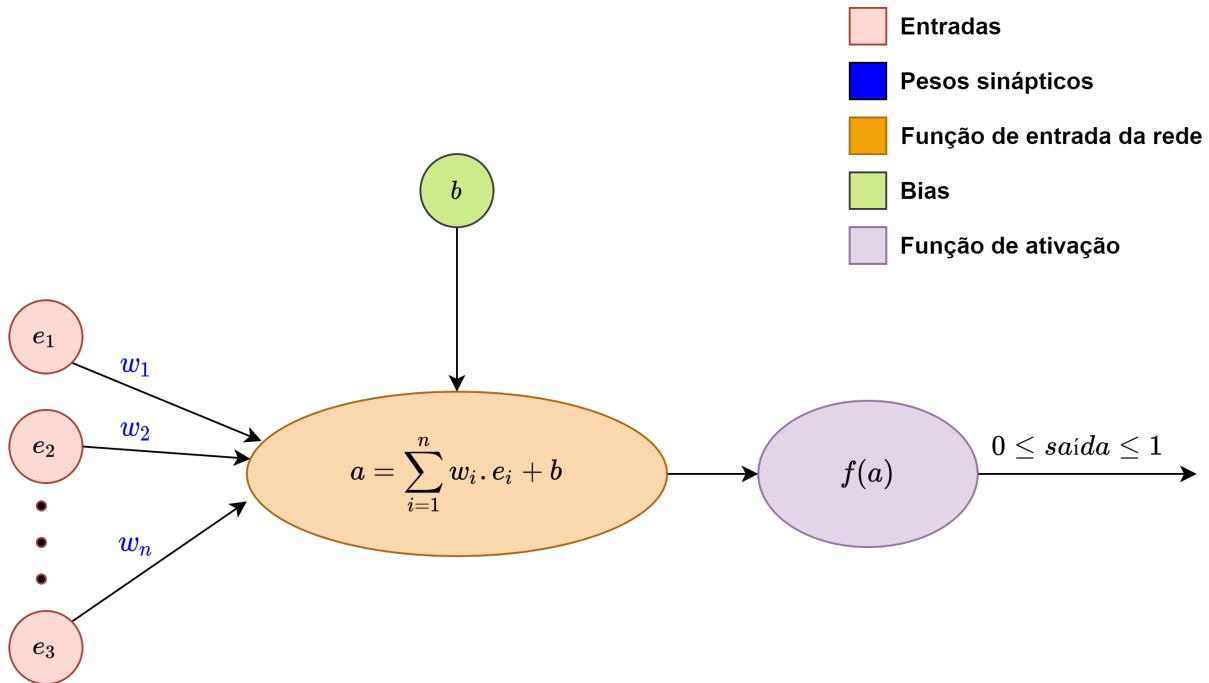
2.3 REDES NEURAIS

O cérebro humano é composto por neurônios, os quais são células que transmitem impulsos nervosos sendo capazes de se comunicarem entre si por meio de sinapses. Em um dado momento, impulsos de entrada são recebidos e processados por um neurônio X, o qual transmitirá, através de neurotransmissores, impulsos nervosos para outro neurônio Y, e isso se repercute criando uma cadeia de informações em uma rede de neurônios, auxiliando na coordenação das atividades necessárias do ser humano (CARVALHO, 2009).

As Redes Neurais Artificiais (RNAs), são modelos de gerenciamento de informações que se baseiam no funcionamento dos sistemas nervosos biológicos do cérebro humano (ABIODUN et al., 2018). Para Haykin, uma RNA é considerada uma máquina capaz de realizar uma determinada tarefa de interesse tão bem quanto o cérebro humano (HAYKIN, 2010).

O neurônio de uma RNA funciona basicamente no mesmo raciocínio de um neurônio biológico, onde os valores de entrada (*inputs*) correspondem aos impulsos nervosos, que são enviados para os nós da RNA, correspondentes aos neurônios, onde são processados e por fim gera-se um resultado final, o qual pode ser interpretado como uma reação aos estímulos de entrada (VAZ, 2018).

Figura 7 – Neurônio fundamental de uma RNA.

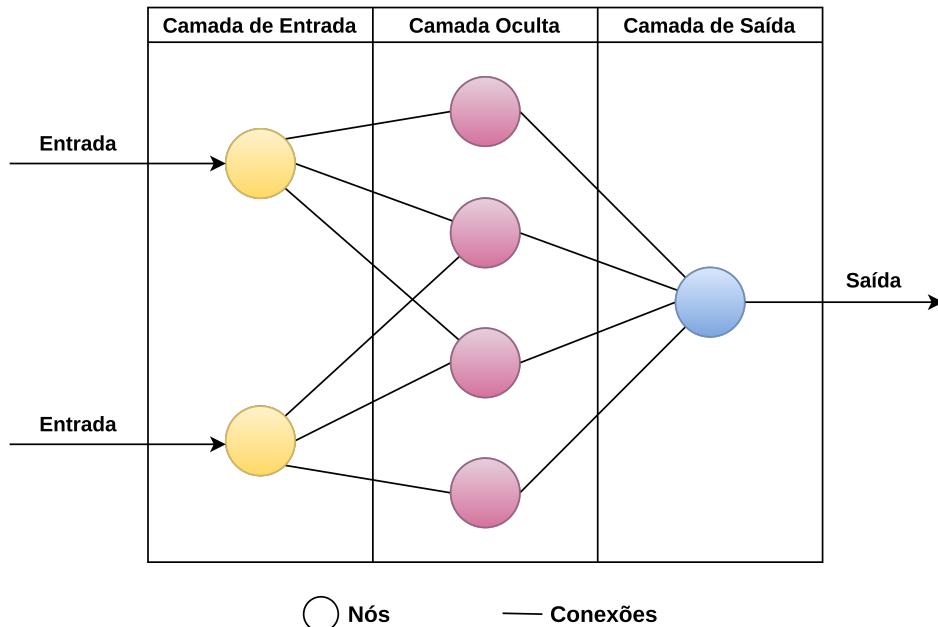


Fonte: Adaptada de (VAZ, 2018).

Na Figura 7 é apresentada a unidade de processamento essencial de uma RNA, a qual foi proposta em 1943, chamada neurônio de McCulloch e Pitts (MCCULLOCH; PITTS, 1943). Como é destacado na figura, este neurônio é composto por cinco componentes principais, onde o fluxo de funcionamento se inicia a partir da entrada de dados, semelhantemente aos estímulos no neurônio biológico. Essas entradas são combinadas com os pesos, que tem o papel de atribuir relevância às entradas impactando-as de acordo com a determinada tarefa que o algoritmo está aprendendo. Após isto, os resultados são somados entre si e ainda é adicionada uma constante denominada bias (b). Por fim, a soma é dada como entrada da função de ativação que irá definir os valores de saída entre 0 e 1 (NICHOLSON, 2020).

A estrutura simples de uma RNA pode ser visualizada também em cinco partes principais como apresentado na Figura 8. Uma camada de entrada, onde os dados são inseridos, a qual se comunica com a camada oculta. Uma camada oculta, onde o processamento acontece através de sistema de conexões ponderadas. Uma camada de saída, que se comunica com a camada oculta e gera os resultados finais. Os nós, que representam os neurônios artificiais, discutido anteriormente, presentes dentro das camadas. E por fim as conexões, que como o nome sugere, conectam os nós entre si, consequentemente formando uma rede de neurônios, ou seja, de nós interconectados (CHEN, 2020).

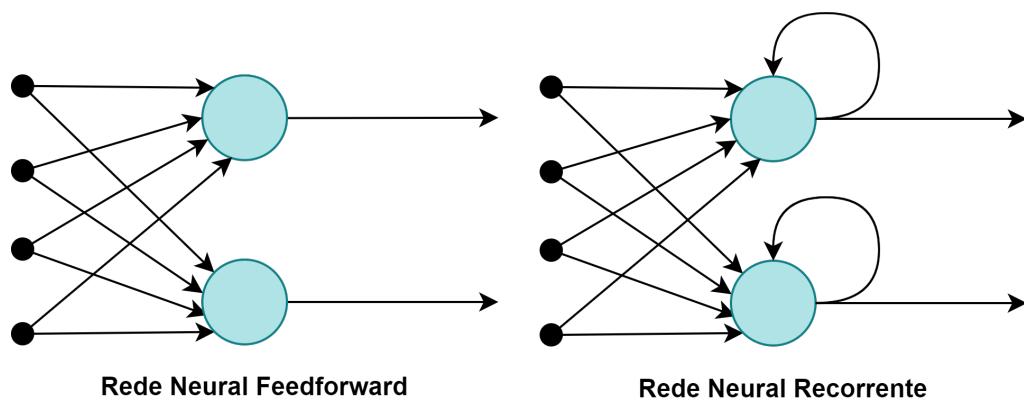
Figura 8 – Estrutura de uma RNA simples.



Fonte: Adaptada de (CHEN, 2020).

As RNAs podem ser classificadas de duas formas em relação a propagação dos dados pela rede. Quando o fluxo de dados de um nó de uma camada sempre envia as informações para o nó da próxima camada, sempre seguindo em frente, então a RNA é dita *feedforward*. Em contrapartida, caso há algum loop no fluxo de dados que percorre a rede, então ela denominada rede recorrente (*com feedback*) (KRÖSE et al., 1993).

Figura 9 – Tipos de RNAs quanto ao fluxo de dados.



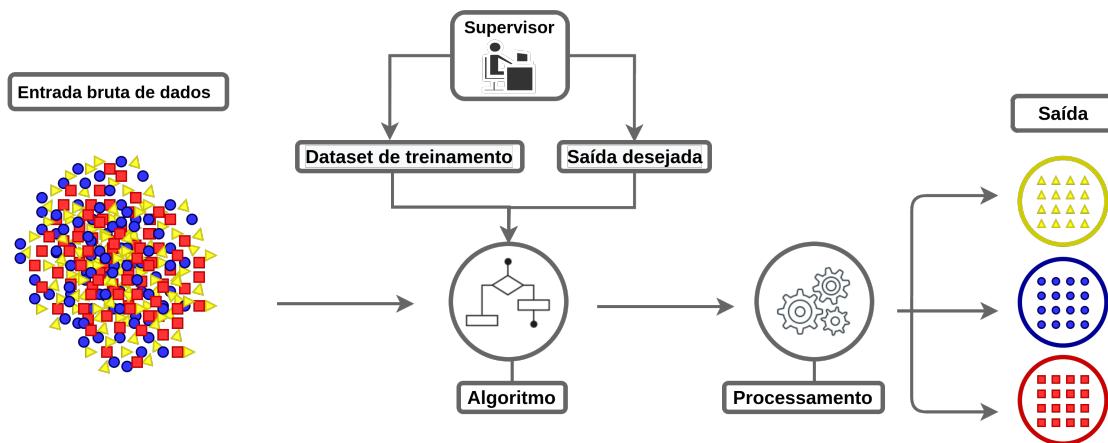
Fonte: Adaptada de (ACADEMY, 2021).

A etapa de treinamento de uma RNA é um importante passo para encontrar um conjunto de pesos necessários para que a partir de um conjunto de entradas possa-se produzir um conjunto de saídas desejado e coerente de acordo com o problema específico que se espera resolver (BROWNLEE, 2019c). Nesse sentido, existem diversos paradigmas de

treinamento/aprendizado. Neste TCC caracterizamos dois tipos: paradigma baseado no aprendizado supervisionado e paradigma baseado no aprendizado não-supervisionado.

O aprendizado supervisionado, também chamado aprendizado associativo, é caracterizado por utilizar técnicas onde a rede recebe um conjunto de dados rotulados para que aprenda a partir deles, ou seja, há uma dependência da intervenção humana no processo de aprendizado (KRÖSE et al., 1993).

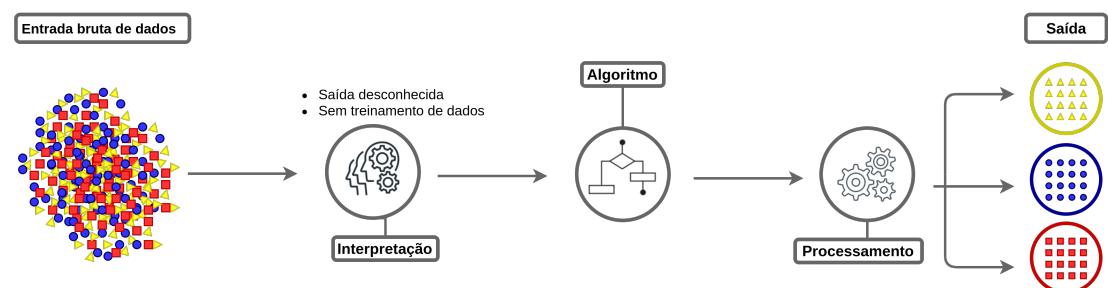
Figura 10 – Processo de aprendizado supervisionado de uma RNA.



Fonte: Adaptada de (MEZIC, 2020).

No aprendizado não-supervisionado, também denominado aprendizado auto-associativo, a rede neural aprende através de dados não rotulados, tendo que desenvolver a própria representação de estímulos de entrada para detectar as semelhanças entre os dados (KRÖSE et al., 1993).

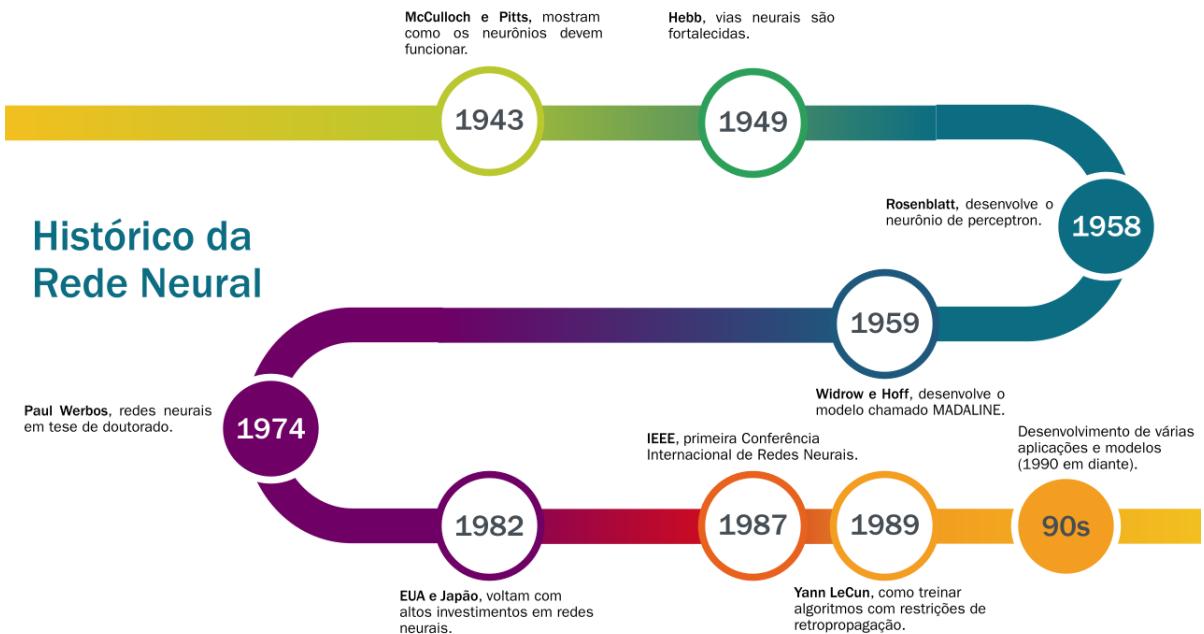
Figura 11 – Processo de aprendizado não-supervisionado de uma RNA.



Fonte: Adaptada de (MEZIC, 2020).

Do ponto de vista histórico, os estudos de RNAs iniciam-se a partir da década de 1940 (EDUCATION, 2020), como ilustrado na Figura 12.

Figura 12 – Breve histórico das RNAs.



Fonte: Adaptada de (STRACHNYI, 2019).

Apesar de o estudo sobre cérebro humano ser mais antigo, a primeiro aparecimento de RNAs surgiu em 1943, a partir do artigo *A logical calculus of the ideas immanent in nervous activity*, que tinha como objetivo entender como o cérebro humano poderia produzir padrões complexos através de neurônios, modelando assim uma rede neural simples por meio de circuitos elétricos (MCCULLOCH; PITTS, 1943).

O livro *The Organization of Behavior* aponta que as vias neurais são fortalecidas conforme cada vez mais são usadas, corroborando com o conceito de neurônios proposto por McCulloch e Pitts (HEBB, 1949).

No artigo *The perceptron: a probabilistic model for information storage and organization in the brain*, Frank Rosenblatt desenvolve o *perceptron*, capaz de classificar um conjunto de entradas de valor contínuo em uma de duas classes a partir da soma ponderada das entradas, introduzindo assim pesos na equação original (ROSENBLATT, 1958).

Em 1959 se deu a aplicação da primeira RNA em um problema real (WIDROW; LEHR, 1990). Tal rede, conhecida como MADALINE é considerada um filtro adaptativo capaz de remover ecos de linhas telefônicas e é utilizada comercialmente até hoje.

Com a acentuada diminuição das pesquisas dentro da área de inteligência artificial, incluindo as redes neurais, houve pequenas contribuições ao longo das décadas de 1960 e 1970.

Em 1982, em uma conferência entre EUA e Japão voltada para discutir assuntos referentes a redes neurais cooperativas e competitivas, o Japão anunciou um aumento de investimentos

em pesquisas na área. Os EUA decidiram também aumentarem os investimentos na área (AMARI; ARBIB, 2013).

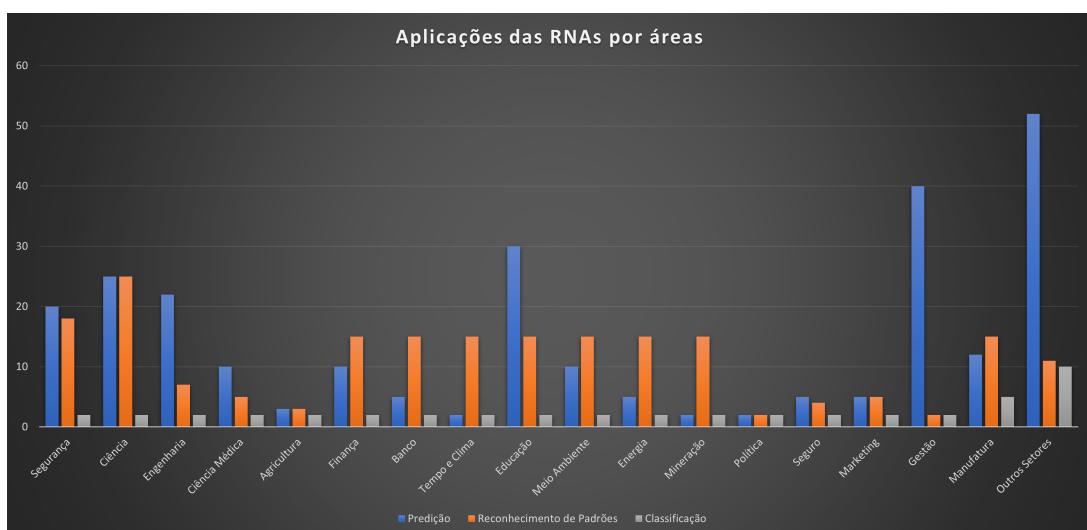
Em 1987 foi realizada a primeira conferência internacional de Redes Neurais promovida pelo Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE), a qual teve milhares de participantes (IEEE, 1987). Dois anos depois, em 1989, Yann LeCun e outros pesquisadores publicaram o artigo *Backpropagation applied to handwritten zip code recognition*, no qual mostram como treinar algoritmos com a utilização de restrições na retropropagação integrada a uma arquitetura de RNA (LECUN et al., 1989).

Da década de 1990 em diante houve o desenvolvimento de diversas aplicações e modelos voltado para a área de redes neurais, como um sistema de rastreamento de gestos de mãos em uma sequência de quadros com alta performance (NOWLAN; PLATT, 1995), a invenção da arquitetura de rede neural recorrente chamada *Long Short-Term Memory (LSTM)* (HOCHREITER; SCHMIDHUBER, 1997) e a aplicação de redes convolucionais para reconhecimento de documentos (LECUN et al., 1998).

As RNAs são comumente utilizadas para resolver problemas complexos auxiliando tomadas de decisões por diferentes entes do mercado. Dentre suas aplicações encontradas no mercado destacamos: otimização de logística em sistemas de transporte, previsão de demanda de energia e carga elétrica, detecção de fraudes na área financeira e literária, reconhecimento de voz e afins.

Na Figura 13 é apresentado um relacionamento entre diversas áreas de aplicações das RNAs. É possível notar que mesmo com diversos campos distintos, as RNAs podem ser aplicadas em praticamente qualquer área do conhecimento humano (ABIODUN et al., 2018).

Figura 13 – Aplicações das RNAs por áreas.



Fonte: Adaptada de (ABIODUN et al., 2018).

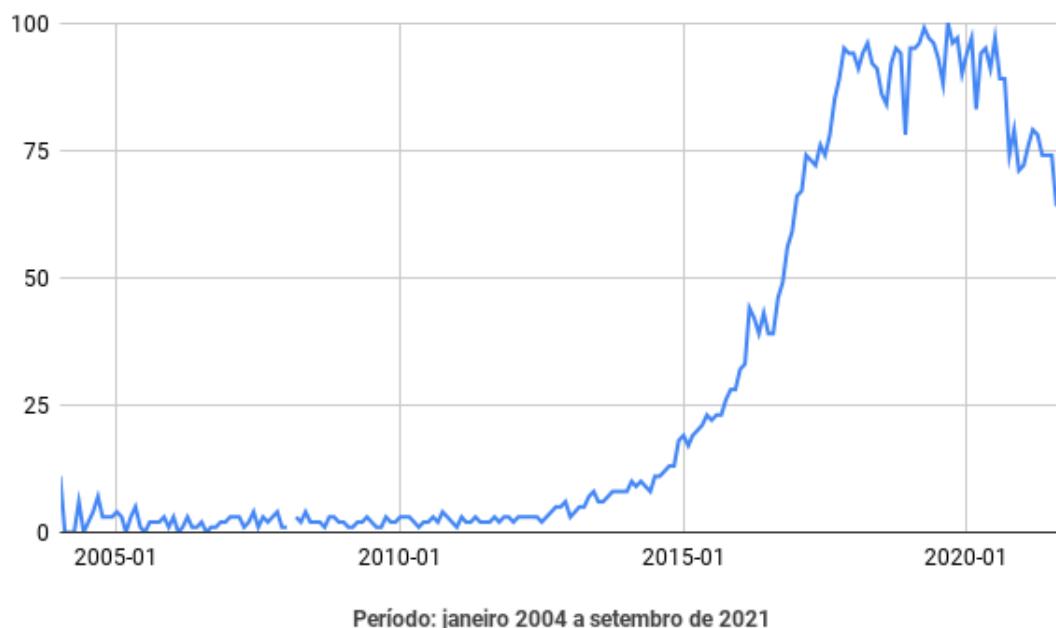
Este vasto número de campos em que as RNAs tem sido ampliado com grande velocidade devido ao advento do aprendizado profundo atrelado ao crescimento do poder computacional ao longo dos anos.

2.3.1 Redes Neurais Profundas

Como uma sub-área da inteligência artificial, mais especificamente do aprendizado de máquina, o aprendizado profundo está no centro das atenções nos dias atuais e adentra vários campos de inovação, tais como: guiagem autônoma, reconhecimento de fala e imagem e processamento de linguagem natural (SAS, 2021).

Tanto empresas de pequeno quanto de médio ou grande porte têm buscado os resultados benéficos que o aprendizado profundo em suas diferentes arquiteturas de redes neurais proporcionam, como diversas pesquisas apontam. Os financiamentos de empresas *startups* em todo mundo foram em torno de 26 bilhões de dólares entre os anos de 2014 e 2019. A Figura 14 apresenta o gráfico da busca pelo termo *deep learning* no *Google* desde 2004, indicando o crescimento no interesse pela temática com o decorrer dos anos e o seu interesse atual. Na nossa opinião a pequena queda de interesse observada a partir de 2020 ocorreu devido à pandemia COVID-19.

Figura 14 – Interesse pelo termo *Deep Learning*.

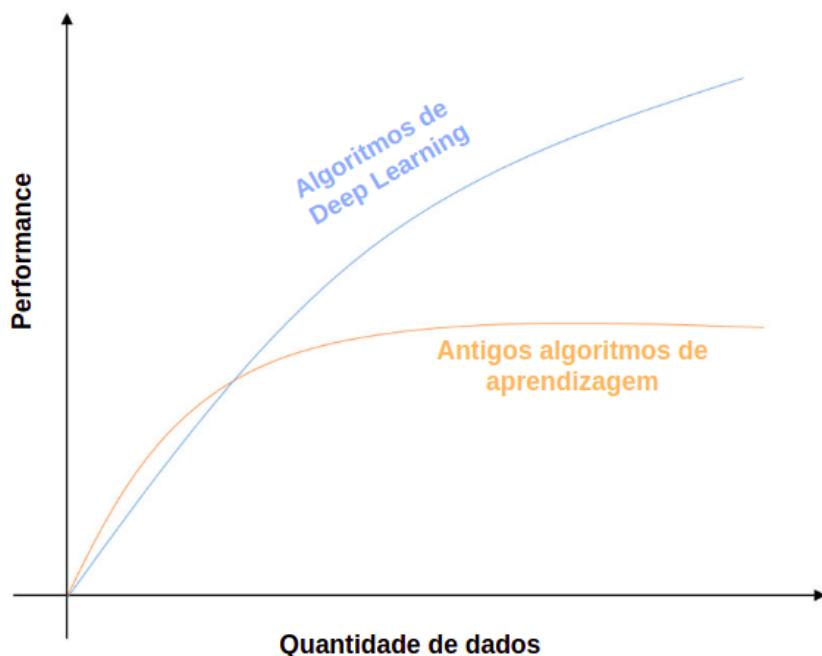


Fonte: Dados obtidos pela ferramenta *Google Trends* <www.google.com/trends>. Acesso em: 22 de set. 2021.

O aprendizado profundo tomou forma a partir da década 1980, porém adentrou de forma mais evidente na áreas científica e tecnológica nos últimos anos, principalmente depois dos

anos 2000. O grande volume de dados disponíveis e gerados atualmente aliado ao elevado poder computacional comparado ao da época, têm permitido o aumento de velocidade no treinamento de algoritmos, bem como a melhoria de performance (BASHAR, 2019). A Figura 15 reflete isto, mostrando que quanto maior a quantidade de dados ou tamanho do modelo computacional os algoritmos que envolvem aprendizado profundo se sobressaem em relação aos demais.

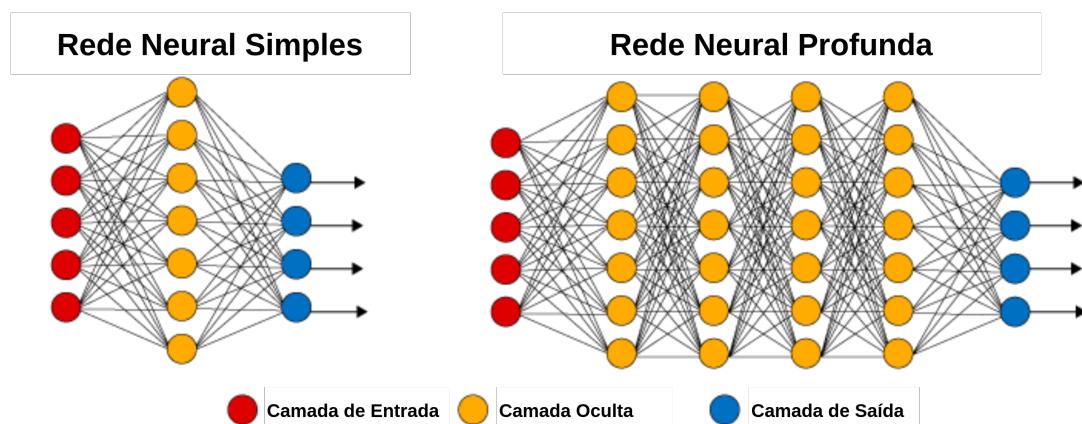
Figura 15 – Performance dos algoritmos de aprendizado profundo em relação à quantidade de dados.



Fonte: Adaptada de (ALOM et al., 2019).

As redes neurais profundas são similares às redes neurais rasas em sua forma de operação, porém uma das características diferenciais é o número de camadas ocultas, que pode chegar a milhares, em alguns casos (ACADEMY, 2021).

Figura 16 – Estruturas de RNAs rasa e profunda.



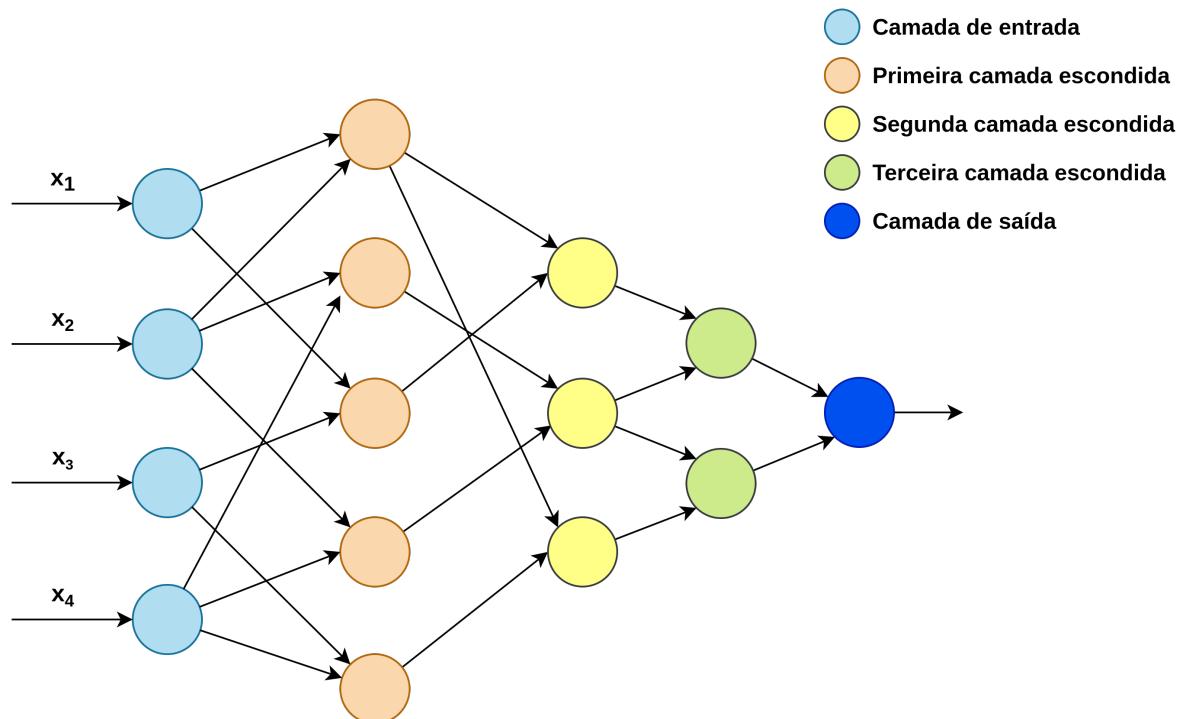
Fonte: Adaptada de (VÁZQUEZ, 2017).

Existem diversos pontos positivos na utilização das redes neurais profundas, como em casos de encontrar bons resultados de análises precisas e confiáveis em algumas tarefas de identificação de anomalias e padrões em grande volume de dados brutos. Além disso, também é capaz de descobrir informações não pretendidas, mas relevantes para tomadas de decisões a partir da base de dados de entrada.

Segundo Yulia Gavrilova (GAVRILOVA, 2020) a *Amazon* possui cerca de 300 milhões de usuários e mais de 560 milhões de itens em seu web site voltado para e-commerce, sendo assim impossível para dezenas ou até mesmo centenas de contadores humanos gerenciar e rastrear volumes gigantescos de transações sem nenhuma espécie de ferramenta que use a inteligência artificial, as quais geralmente envolvem o aprendizado profundo.

A primeira arquitetura que utiliza aprendizado profundo ocorreu em 1965, quando Ivakhnenko e Lapa propuseram uma rede neural que possuía várias camadas não lineares com modelos finos e profundos com funções de ativação polinomial (DETTMERS, 2015). A Figura 17 mostra a estrutura básica da rede proposta, que contém um processo de aprendizado que é lento e manual, onde em cada camada são selecionados os melhores recursos através de métodos estatísticos, encaminhando-os para próxima camada até o ponto que a rede se estreita o máximo possível, indicando que nenhuma melhoria adicional pode ser mais alcançada, mesmo com adição de novas camadas (ACADEMY, 2021).

Figura 17 – Arquitetura da primeira rede neural profunda conhecida treinada por Alexey Grigorevich Ivakhnenko.



Fonte: Adaptada de (ACADEMY, 2021).

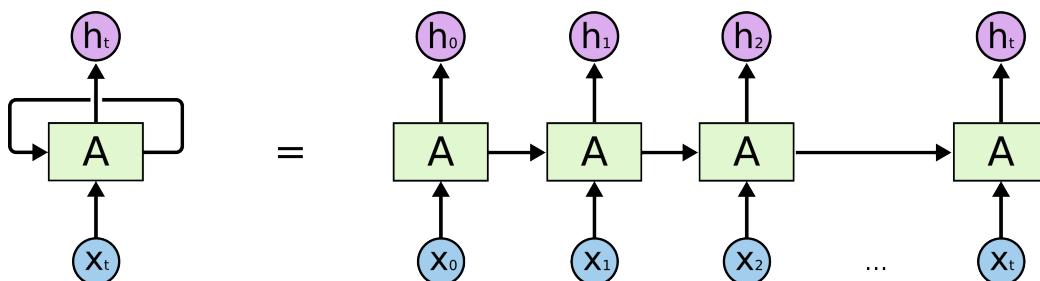
Ao longo dos anos, novos tipos de redes neurais para resolver problemas complexos usando aprendizado profundo (PAI, 2020). Neste TCC será utilizada a rede neural recorrente chamada *Long Short-Term Memory (LSTM)* em conjunto com uma arquitetura de *Rede Neural Siamesa (RNS)*.

2.3.1.1 Redes Neurais *Long Short-Term Memory*

Uma das tarefas mais comuns do cérebro humano está na persistência do acesso a informações. Ao assistir um filme, por exemplo, geralmente não decoramos todo o roteiro ou cada palavra que foi dita/ouvida, mas sim nos principais acontecimentos. O mesmo ocorre ao lermos um livro ou este TCC. Pesquisas neurocientíficas têm apontado que o nosso cérebro atrela a compreensão do significado de cada palavra que se está lendo/ouvindo é baseada na compreensão das palavras com ela correlacionadas, ou seja, que nós assimilamos informações e aprendemos a partir de uma persistência na análise de como uma palavra se relaciona com as vizinhas.

Com a ascensão do aprendizado profundo, tornou-se possível desenvolver estratégias que permitem levar em conta informações de eventos anteriores através das redes neurais recorrentes (RNNs) para a análise de um evento dito atual (KUMAR; GOOMER; SINGH, 2018). Apontando nesse sentido, as redes neurais recorrentes caracterizam-se por apresentarem *loops*, permitindo a ela persistir na análise de informações mantendo uma espécie de memória com base de histórico (HUANG; XU; YU, 2015). Como ilustrado na Figura 18, esses *loops* fazem com que uma RNN possa ser vista como uma classe de várias cópias “desdobradas” dela mesma em retroalimentação, cada qual envia informações para uma sucessora formando uma espécie “cadeia de redes” (OLAH, 2015).

Figura 18 – Rede neural recorrente básica “desdoblada”.



Fonte: (OLAH, 2015).

As RNNs foram introduzidas na década de 1970 e foram popularizadas a partir da década de 1980. Formam um dos tipos de redes neurais profundas mais bem sucedidas em tarefas que envolvem processamento de linguagem natural e reconhecimento de fala (HUANG; XU; YU, 2015).

Uma das primeiras redes recorrentes introduzidas na literatura foi a rede de Hopfield,

concebida por W. Little em 1974 (LITTLE, 1974) e popularizada por John Hopfield em 1982, através do artigo *Neural networks and physical systems with emergent collective computational abilities* (HOPFIELD, 1982). Nele, Hopfield demonstra um sistema binário de memória associativa capaz de aprender múltiplos padrões.

Alguns anos depois, em 1986, David Rumelhart e outros pesquisadores publicaram o artigo em que muitas redes recorrentes se basearam, intitulado *Learning representations by back-propagating errors* (RUMELHART; HINTON; WILLIAMS, 1986). Nele é realizada a aplicação do algoritmo de retropropagação em redes neurais de várias camadas, ou seja, em redes profundas.

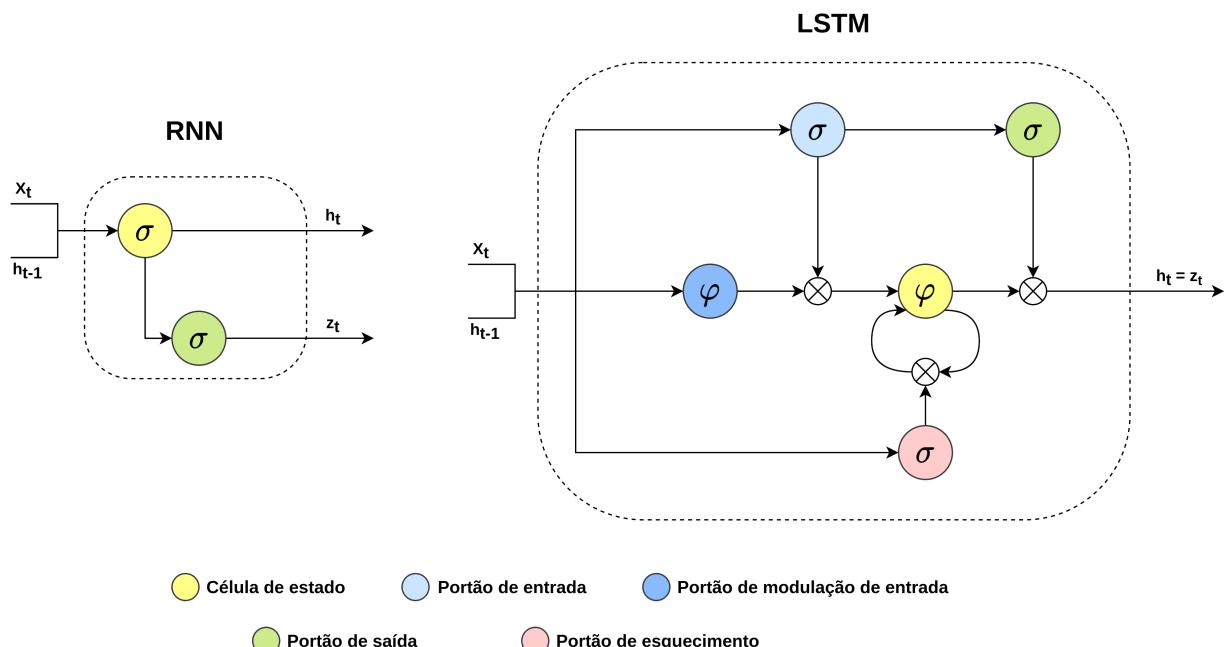
Nos anos posteriores, alguns trabalhos referentes a RNNs obtiveram destaque na comunidade acadêmica. Um exemplo é caso dos algoritmos de aprendizado para redes recorrentes totalmente conectadas para melhoria no aprendizado de tarefas complexas apresentado no artigo *A learning algorithm for continually running fully recurrent neural networks* (WILLIAMS; ZIPSER, 1989). Outro destaque são as redes de Elman e Jordan apresentadas no artigo *Finding structure in time* (ELMAN, 1990). Ambas possuem o mesmo esquema de treinamento e foram aplicadas para reconhecimento de sequências e processamento de linguagem natural.

Todavia, na primeira metade da década de 1990 foram relatadas dificuldades por parte das redes recorrentes durante o aprendizado de longas sequências de dados, o qual também é comumente tratado como *problema da dependência de longo prazo*. Os autores dos artigos *Untersuchungen zu dynamischen neuronalen Netzen* (HOCHREITER, 1991) e *Learning long-term dependencies with gradient descent is difficult* (BENGIO; SIMARD; FRASCONI, 1994) abordaram com profundidade este problema, concluindo que a arquitetura da rede recorrente básica retarda o aprendizado à medida que a lacuna entre informações anteriores desejadas e o ponto de exigência atual aumenta em grande proporção, ocasionando problema de gradiente, ou seja, sua explosão ou o decaimento exponencial do erro de retropropagação, conforme o aumento na duração das dependências exigidas.

Foi então que em 1997, Hochreiter e Schmidhuber introduziram uma rede especial derivada de RNN clássica, denominada *Long short-term memory*, a qual foi projetada para resolver o problema de dependência de longo prazo das redes recorrentes (HOCHREITER; SCHMIDHUBER, 1997). Uma das principais características da *Long short-term memory* (LSTM) é guardar e lembrar informações de um “longo” período sequencial. A LSTM se torna uma escolha natural de uso caso as dependências de informações anteriores afetem diretamente a precisão e qualidade de aprendizado do modelo (KUMAR; GOOMER; SINGH, 2018). Além disso, as LSTMs performam bem em uma considerável gama de problemas e, por isso, são amplamente utilizadas em várias tarefas (OLAH, 2015).

A camada oculta das redes LSTM são compostas por um conjunto de blocos conectados de maneira recorrente, os quais são conhecidos como blocos de memória. Os quatro componentes fundamentais das LSTMs estão contidos em cada um dos blocos, sendo eles: célula de estado (*cell state*) e os portões (*gates*) de esquecimento, entrada e saída (GRAVES; FERNÁNDEZ; SCHMIDHUBER, 2005). Na Figura 19 é apresentada a estrutura de uma célula LSTM em comparação a uma estrutura de RNN clássica. Diferentemente de suas predecessoras, a LSTM consegue armazenar informações na célula de estado e manipular o fluxo delas através dos portões, mantendo assim somente informações consideradas necessárias durante o processo de treinamento (SAK; SENIOR; BEAUFAYS, 2014).

Figura 19 – Estruturas das células de uma RNN clássica e de uma LSTM.

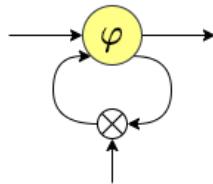


Fonte: Adaptada de (RAVAL, 2017).

Os portões das LSTMs utilizam duas funções de ativação durante a manipulação das informações na rede, as quais são: *tanh* e a sigmóide. A função *tanh* gera valores de saída de -1 a 1. Já a função sigmóide gera valores de 0 a 1, que é normalmente empregado para determinar quais dados são relevantes manter na rede e quais são necessários esquecer (PHI, 2018).

A célula de estado pode ser considerada como o componente mais importante da LSTM, pois ela desempenha a função de memória de longo prazo, armazenando as informações mais relevantes para o treinamento da rede. A mudança de estado da célula é provocada tanto pelo portão de esquecimento, removendo informações, quanto pelo portão de entrada, adicionando informações. As setas em formato de laços exibidas na Figura 20, revelam a natureza recursiva da célula de estado, possibilitando que informações de intervalos anteriores sejam armazenadas nela (KANG, 2017).

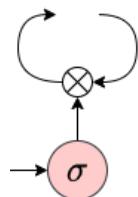
Figura 20 – Célula de estado da LSTM.



Fonte: Adaptada de (RAVAL, 2017).

A Figura 21 apresenta o portão de esquecimento, que é responsável por decidir quais informações devem ser retidas ou descartadas da célula de estado. Isto é determinado a partir do resultado do cálculo das informações do estado oculto anterior e da entrada ponderada atual aplicadas na função sigmóide. Se o resultado for 1 a informação é retida, caso contrário é desconsiderada (KANG, 2017; PHI, 2018).

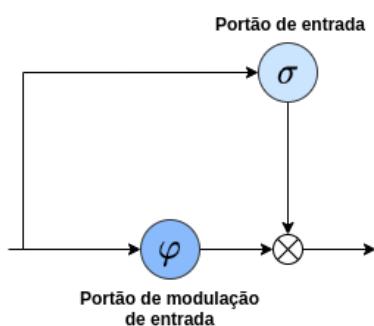
Figura 21 – Portão de esquecimento da LSTM.



Fonte: Adaptada de (RAVAL, 2017).

A Figura 22 apresenta o portão de entrada, que tem o papel de definir quais informações relevantes serão retidas na célula de estado. Para que isto ocorra, as informações do estado oculto anterior e da entrada atual devem passar pelas duas funções de ativação apresentadas anteriormente: sigmóide, aplicada no portão de entrada, e *tahn*, aplicada no portão de modulação de entrada para balancear a rede. Os resultados das saídas das duas funções são multiplicados, e a saída da função sigmóide determina quais informações são importantes reter da saída da função *tahn* (KANG, 2017; PHI, 2018).

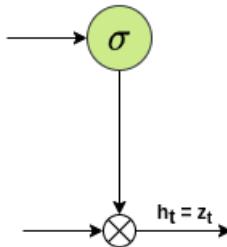
Figura 22 – Portão de entrada da LSTM.



Fonte: Adaptada de (RAVAL, 2017).

Por fim, na Figura 23 é apresentado o portão de saída. Ele determina qual deve ser o próximo estado oculto. Se processo de operação inicia-se a partir da aplicação das informações do estado oculto anterior e da entrada atual na função sigmóide, e logo após, a aplicação das informações da célula de estado na função *tanh*. Os resultados das saídas das funções são então multiplicadas com o fim de determinar quais informações o estado oculto deve carregar para o próximo bloco de memória (KANG, 2017; PHI, 2018).

Figura 23 – Portão de saída da LSTM.



Fonte: Adaptada de (RAVAL, 2017).

Algumas das principais aplicações que incluem as LSTMs incluem: máquinas de tradução (CUI; WANG; LI, 2015; REN, 2020), *chatbots* (XU et al., 2017; SU et al., 2017), geração de textos (PAWADE et al., 2018; SANTHANAM, 2020), modelagem de linguagem (SUNDERMEYER; SCHLÜTER; NEY, 2012; SUNDERMEYER; NEY; SCHLÜTER, 2015), determinação de legendas de imagens (SOH, 2016; TAN; CHAN, 2019) e reconhecimento de fala (GRAVES; JAITLEY; MOHAMED, 2013; HAN et al., 2017; KIM; EL-KHAMY; LEE, 2017; SHEWALKAR, 2019).

A aplicação desenvolvida apresentada no artigo *Application of LSTM neural networks in language modelling* (SOUTNER; MÜLLER, 2013), apresenta uma rede LSTM com extensões para modelagem de linguagem voltada para chamadas telefônicas tchecas. Foi constatado melhorias consideráveis no sistema de reconhecimento de fala através dos resultados levantados.

Um problema identificado no artigo *Soft sensor development and applications based on LSTM in deep neural networks* indica que redes profundas não são eficazes em processos químicos dinâmicos, não lineares e complexos. Uma rede LSTM foi aplicada em sensores suaves para tratar este problema. Os experimentos realizados forneceram bons resultados (KE et al., 2017).

O grande volume de dados gerados em diferentes canais acaba se tornando um empecilho para criação de preditores relevantes e eficientes para prever o comportamento do cliente, especialmente se é utilizado modelos de previsões tradicionais. O artigo *LSTM response models for direct marketing analytics: Replacing feature engineering with deep learning* mostra que as redes LSTMs possuem alta precisão na tarefa de previsão do comportamento

do cliente, com resultados superiores comparados aos dos modelos tradicionais (SARKAR; BRUYN, 2021).

2.4 REDES NEURAIS SIAMESAS

Atualmente existem diversas aplicações da área de inteligência artificial que giram em torno de identificar se dois ou mais objetos são similares (CHICCO, 2021). Como no caso de aplicações que desempenham tarefas voltadas para o processamento de linguagem natural (PLN), tais como: recuperação de informações, summarização de textos, extração de informações, identificação de autoria e afins. Para que essas aplicações consigam efetuar suas funções com boa performance, há a dependência de um componente essencial chamado de semântica de similaridade textual (STS), capaz de determinar o quanto “próximos” são instâncias de sentenças de texto através da medição de similaridade entre as mesmas (PONTES et al., 2018). Diferentes meios são adotados para mensurar a STS e um deles é a utilização das redes neurais siamesas (RNS), as quais são um tipo especial de RNA especializada em medir semelhanças (CHICCO, 2021).

A introdução das redes siamesas na literatura se deu em 1993, onde Bromley, LeCun e outros pesquisadores utilizaram a arquitetura da rede criada para resolver problemas de verificação de assinaturas. O artigo *Signature verification using a "siamese" time delay neural network* mostrou que a solução proposta teve uma eficiência de detecção de assinaturas em torno de 95%, provocando uma grande repercussão na época (BROMLEY et al., 1993).

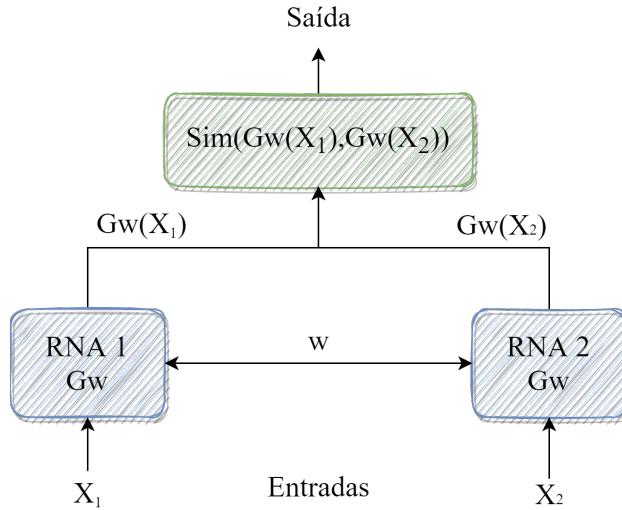
Em 2005, novamente LeCun e outros pesquisadores formalizam no artigo *Learning a similarity metric discriminatively, with application to face verification* a arquitetura da RNS, aplicando-a em conjunto com uma rede CNN para atuar na tarefa de verificação facial (CHOPRA; HADSELL; LECUN, 2005). Com o passar dos anos, as redes siamesas foram ganhando cada vez mais espaço, sendo aplicadas com sucesso em várias tarefas que envolvem processamento de linguagem natural e visão computacional.

As RNS são uma classe especial das RNAs, as quais são caracterizadas por possuírem duas ou mais sub-redes idênticas com as mesmas configurações, parâmetros e pesos compartilhados. Durante as atualizações dos parâmetros há o espelhamento para todas as suas sub-redes (RANASINGHE; ORASAN; MITKOV, 2019; J, 2020). Quanto às estratégias de treinamento empregadas nas redes siamesas, geralmente se utiliza métricas de aprendizado de similaridade não linear (BERLEMONT et al., 2018; SOUZA et al., 2019).

A Figura 24 apresenta uma arquitetura padrão de uma rede siamesa, em que as duas sub-redes iguais são ligadas por camadas compartilhando pesos entre si. Tanto no processo

de treinamento quanto nos testes de predição, a rede é alimentada por pares de entrada no intuito de aproximar pares de uma mesma classe e distanciar pares de classes diferentes.

Figura 24 – Arquitetura padrão de uma rede neural siamesa (RNS).



Fonte: Adaptada de (SOUZA et al., 2019).

A rede gera duas saídas $G_w(X_1)$ e $G_w(X_2)$ as quais são comparadas usando uma medida de similaridade Sim , e fornecem a saída $\text{Sim}(G_w(X_1), G_w(X_2)) \in [0, 1]$. Quanto mais próximo de 1 for o valor de $\text{Sim}(G_w(X_1), G_w(X_2))$, maior será a similaridade entre os pares (SOUZA et al., 2019).

As redes siamesas apresentam boa performance em problemas que envolvem: verificação facial (CUI et al., 2019; LU et al., 2021), verificação de assinaturas (JAGTAP et al., 2020; GHOSH et al., 2021), semelhança de imagens (MELEKHOV; KANNALA; RAHTU, 2016; WIGGERS et al., 2019) e semelhança de sentenças (YIH et al., 2011; MUELLER; THYAGARAJAN, 2016; ICHIDA; MENEGUZZI; RUIZ, 2018; CHI; ZHANG, 2018).

O artigo *Learning text similarity with siamese recurrent networks* (NECULOIU; VERS-TEEGH; ROTARU, 2016) apresenta um modelo de rede neural siamesa combinado com LSTM bidirecionais para aprendizado de uma métrica de similaridade em sequências de caracteres de comprimento variável. Foram obtidos resultados que levam em consideração diferentes tipos de entradas relativas à variação ortográfica, substituição de sinônimos e até mesmo palavras supérfluas com boa performance. Além disso, a arquitetura do modelo proposto escalava de forma natural à medida que o número de classes aumentava.

Pesquisas recentes na área de varejo sugerem que uso de análise de dados funcionais melhoram os perfis de vendas de produtos. Baseado nisto, o trabalho apresentado no artigo *A Siamese Neural Network Application for Sales Forecasting of New Fashion Products Using Heterogeneous Data* propõe uma aplicação capaz de realizar previsões de vendas de longo prazo de novos produtos usando RNSs em combinação com CNNs. Os experimentos

realizados comprovam que a aplicação desenvolvida fornece previsões de vendas satisfatórias dos produtos do mercado varejista (CRAPAROTTA; THOMASSEY; BIOLATTI, 2019).

SHORFUZZAMAN e HOSSAIN (2021), no artigo *MetaCOVID: A Siamese neural network framework with contrastive loss for n-shot diagnosis of COVID-19 patients* propõem uma RNS em combinação com uma CNN que é capaz de agilizar o processo de análise de imagens de raio-X do tórax dos pacientes a fim de detectar automaticamente os casos da doença COVID-19. Os resultados experimentais demonstram que, apesar das limitações de amostras de treinamento, o modelo proposto tem uma precisão em torno de 95,6% e *area under curve* (AUC) de 0,97 na detecção da doença (SHORFUZZAMAN; HOSSAIN, 2021).

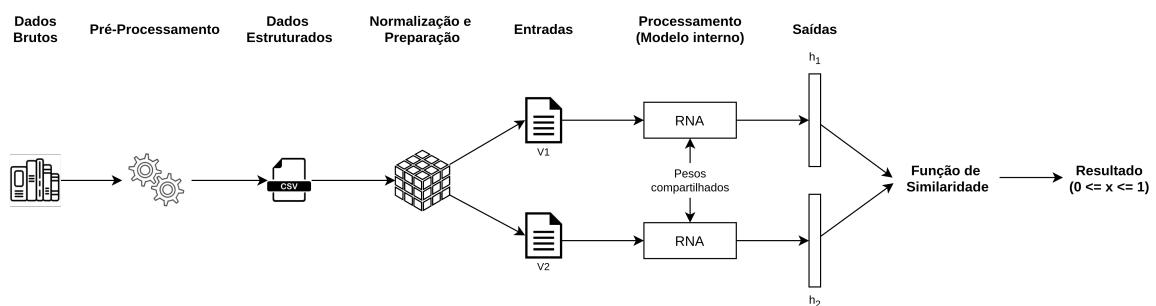
3 MATERIAIS E MÉTODOS

Neste capítulo são apresentados de forma detalhada os recursos e estudos realizados para aplicação da RNA escolhida para conduzir os experimentos computacionais definidos. É importante ressaltar que ao longo das seções deste capítulo e dos posteriores, os termos frase e sentença possuem o mesmo significado e, portanto serão utilizados como sinônimos.

3.1 ESTRUTURA GERAL DOS EXPERIMENTOS COMPUTACIONAIS

A obtenção dos resultados de performance esperados no trabalho através da rede neural siamesa depende de um conjunto de etapas, como é apresentado na Figura 25. Cada uma das etapas possui seus procedimentos e responsabilidades, os quais serão brevemente detalhados nesta seção e nas seções posteriores.

Figura 25 – Fluxo de funcionamento principal da rede siamesa utilizada.



Fonte: Elaborado pelo autor (2021).

A Figura 25 o fluxo de tratamento de informações que direcionou nossa metodologia. Na primeira etapa é realizada a captura dos dados, que neste caso são os textos literários de autores da literatura norte-americana. Em sequência são selecionadas sentenças de textos das obras que passam por um pré-processamento, realizando assim uma filtragem e a normalização dos dados. Cada uma das frases selecionadas são armazenadas e transformadas em linhas em arquivos no formato *comma-separated values* (CSV) no seu processo de estruturação.

Na etapa seguinte é realizada o processo de preparação dos dados de entrada lidos de cada frase dos arquivos CSV, a criação dos vetores de índices, dos *embeddings* das palavras, a definição de hiperparâmetros da rede e a separação dos dados de treinamento e validação das sub-redes, que são alimentadas posteriormente por pares de entradas. Após isto, a rede realiza o processamento dos dados nas camadas escondidas, extraíndo as *features* dos estilos de escritas dos autores selecionados, sendo considerada uma etapa fundamental tanto no processo de treinamento quanto no de teste/predição. Ambos os modelos utilizados das subredes são LSTMs idênticas, as quais são previamente modeladas e configuradas.

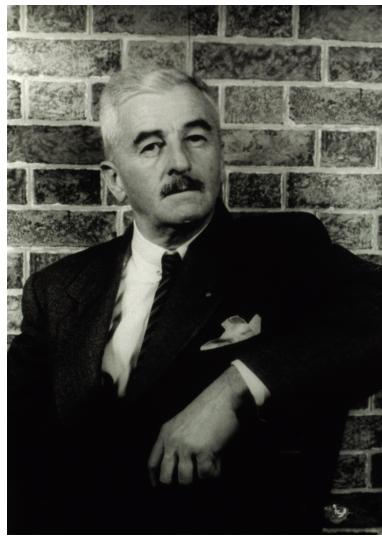
Por fim, as saídas de ambas as sub-redes, que são representações vetoriais das entradas, são submetidas a uma medida de similaridade, que é uma função matemática responsável por realizar o processo de *merge* entre as duas sub-redes utilizadas. O resultado de saída da função determina o quão “próximo” são as saídas por meio de um valor numérico entre 0 e 1 chamado índice de similaridade. Como apresentado no Capítulo 2, a medida de similaridade aplicada para análise da rede siamesa no trabalho em questão é a medida de similaridade de *Manhattan*.

3.2 BASE DE DADOS

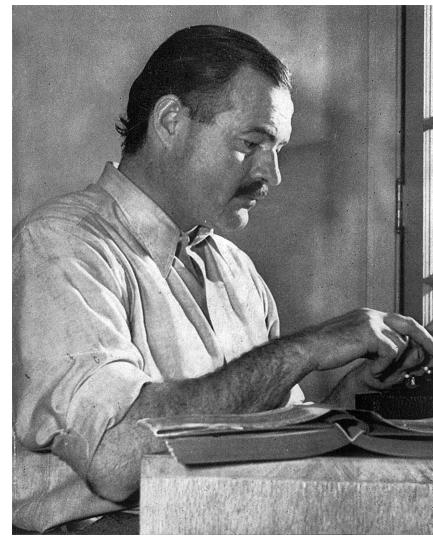
A base de dados do trabalho foi criada a partir de sentenças de textos de livros de autores romancistas renomados da literatura norte-americana. Como mostrado na Figura 26, todos os três autores selecionados produziram grandes trabalhos na área, o que renderam-lhes famosas premiações, como por exemplo o prêmio nobel de Literatura.

Figura 26 – Autores romancistas da literatura norte-americana.

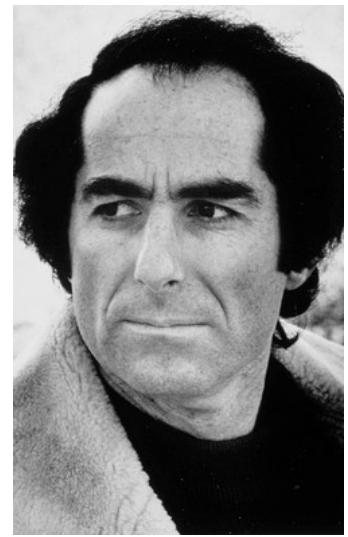
(a) William C. Faulkner



(b) Ernest M. Hemingway



(c) Philip M. Roth



Fonte: Da esquerda para direita as imagens foram retiradas respectivamente de <<https://www.britannica.com/biography/William-Faulkner>>, <<https://www.byanyothernerd.com/2014/08/short-story-228-in-another-country-by.html>> e <https://hudsonvalleyone.com/2018/05/29/philip-roth-is-laid-to-rest-in-annandale>. Acesso em: 14 de jun. 2021.

Para fins investigativos deste TCC, as sentenças selecionadas de textos dos autores, apresentadas na Figura 26, foram divididas em três conjuntos de dados: D_1 , D_2 e D_3 . D_1 corresponde ao conjunto de dados com a aplicação de um pré-processamento textual básico, como será explicado mais à frente neste texto; D_2 deriva de D_1 a partir da remoção de stopwords; D_3 deriva de D_1 pela remoção de stopwords e a aplicação de lematização nas palavras. Os detalhes acerca dessas técnicas de pré-processamento de dados aplicados nos conjuntos de dados D_1 , D_2 e D_3 são apresentadas nas próximas seções.

3.2.1 William C. Faulkner

O escritor norte-americano Willian Cuthbert Faulkner nasceu em 1897 em difícil cenário familiar após trinta anos do fim Guerra Civil Americana. Ele teve que interromper seus estudos acadêmicos em Literatura para se dedicar a trabalhos eventuais. Um deles foi a publicação da sua primeira obra, uma coletânea de poemas chamada *The Marble Faun*, em 1924. Dois anos depois é publicado o seu primeiro romance, intitulado *Soldier's Pay*, lançado graças à influência de seu amigo Sherwood Anderson, que o auxiliou a alavancar na literatura romancista ao longo da sua vida. Suas obras atravessam a narrativa da decadência do sul dos EUA por meio de histórias complexas e desafiadoras. Em 1962, pouco tempo depois de publicar seu último romance, teve problemas cardíacos e acabou falecendo aos 64 anos de idade.

Faulkner foi aclamado pelas suas obras clássicas da literatura, tais como: *The Sound and the Fury* (1929), *Light in August* (1932) e *A Fable* (1954). Ganhou dois prêmios Pulitzer de Ficção (1955 e 1962) e um prêmio Nobel de Literatura (1949). A Tabela 1 apresenta as obras escolhidas do autor Faulkner das quais extraímos as sentenças que compuseram as nossas bases de dados.

Tabela 1 – Quantidade de sentenças por obra selecionada de Faulkner para a composição das nossas bases de dados.

Processo	Obra	Qtd. Sentenças			Total
		D ₁	D ₂	D ₃	
Treinamento/Validação	A Fable (1954)	3932	3918	3917	24000
	Absalom, Absalom! (1936)	2400	2345	2345	
	As I Lay Dying (1930)	2400	2572	2573	
	Intruder in the Dust (1949)	868	765	765	
	Light in August (1932)	2400	2400	2400	
	Sanctuary (1931)	2400	2400	2400	
	Soldiers Pay (1926)	2400	2400	2400	
	The Hamlet (1940)	2400	2400	2400	
	The Mansion (1959)	2400	2400	2400	
	The Reivers (1962)	2400	2400	2400	
Teste	The Sound and the Fury (1929)	200	200	200	200

Fonte: Elaborada pelo autor (2021).

3.2.2 Ernest M. Hemingway

Em 1899, dois anos após o nascimento de Faulkner, nasce também outro importante escritor norte-americano de grande relevância para a Literatura norte-americana: Ernest Miller Hemingway. Apesar de suas participações nas guerras mundiais, trabalhou desde o início da sua carreira como escritor de jornal e revista, o que criou um cenário que permitiu a publicação de seus primeiros contos durante a primeira metade da década de 1920. Fazia parte da comunidade de escritores conhecida como “Geração Perdida” e ganhou fama na

Literatura Romancista com uma das suas primeiras obras intitulada de *The Sun Also Rises* (1926). Sua característica literária é marcada pelo estilo sintético empregado no jornalismo, refletindo em boa parte de suas obras as suas experiências pessoais. A vida conturbada e com problemas psicológicos acumulados desde sua infância o levou ao suicídio em 1961, aos 61 anos de idade.

Suas obras de maior destaque são *For Whom the Bell Tolls* (1940) e *The Old Man and the Sea* (1952), as quais foram de suma importância para que ele ganhasse o Pulitzer de Ficção (1953) e o nobel de Literatura (1954). A Tabela 2 apresenta as obras selecionadas do autor das quais extraímos os dados de treinamento, validação e teste dos nossos modelos.

Tabela 2 – Quantidade de sentenças por obras selecionadas de Hemingway para a composição das nossas bases de dados.

Processo	Obra	Qtd. Sentenças			Total
		D ₁	D ₂	D ₃	
Treinamento/Validação	A Farewell to Arms (1929)	3542	4324	4328	24000
	A Moveable Feast (1964)	1609	1524	1524	
	Across the River and Into the Trees (1950)	2400	2400	2404	
	Death in the Afternoon (1932)	2400	2400	2400	
	For Whom the Bell Tolls (1940)	2400	1969	1965	
	Green Hills of Africa (1936)	2400	2400	2400	
	Men Without Woman (1926)	2385	2217	2216	
	The Garden of Eden (1986)	2400	2400	2400	
	The Old Man and the Sea (1952)	2064	1966	1963	
	The Sun Also Rises (1926)	2400	2400	2400	
Teste	To Have and Have Not (1937)	200	200	200	200

Fonte: Elaborado pelo autor (2021).

3.2.3 Philip M. Roth

Philip Milton Roth é um escritor romancista norte-americano de origem judaica que nasceu em 1933. Com influências dos pensamentos e ideias complexas de Franz Kafka e Sigmund Freud, Roth retrata em suas obras as dificuldades das vidas dos judeus em território norte americano, explorando principalmente os aspectos de identidade pessoal, libido sexual e a autocompreensão. Em 2012, com 79 anos de idade, decidiu-se aposentar da carreira de escritor pouco tempo depois de ganhar seu último prêmio. Morreu em 2018, com 85 anos de ano, vítima de insuficiência cardíaca.

Roth é considerado por vários críticos como um dos escritores romancistas mais bem sucedidos da literatura de língua inglesa, ganhando diversos prêmios ao longo da sua carreira, tais como: Pulitzer de Ficção (1998), PEN/Faulkner de Ficção (1994, 2001, 2007), Franz Kafka (2001), International Man Booker (2011), entre outros. Suas obras mais conhecidas e relevantes foram criadas durante a segunda metade do século XX. A Tabela 3 apresenta as obras selecionadas do autor das quais extraímos os dados de treinamento, validação e teste dos nossos modelos.

Tabela 3 – Quantidade de sentenças por obras selecionadas de Roth para a composição das nossas bases de dados.

Processo	Obra	Qtd. Sentenças			Total
		D_1	D_2	D_3	
Treinamento/Validação	American Pastoral (1997)	2981	3382	3384	24000
	Everyman (2006)	1847	1662	1660	
	Goodbye, Columbus (1959)	2000	2000	2000	
	I Married a Communist (1998)	2000	2000	2000	
	Operation Shylock (1983)	2000	2000	2000	
	Our Gang (1971)	1360	1256	1256	
	Patrimony: A True Story (1991)	1812	1700	1700	
	Portnoy's Complaint (1969)	2000	2000	2000	
	Sabbaths Theater (1995)	2000	2000	2000	
	The Counterlife (1986)	2000	2000	2000	
	The Professor of Desire (1977)	2000	2000	2000	
	When She Was Good (1967)	2000	2000	2000	
Teste	My Life As A Man (1974)	200	200	200	200

Fonte: Elaborado pelo autor (2021).

A Tabela 4 mostra a summarização do total de sentenças extraídas das obras literárias de todas as três tabelas apresentadas anteriormente para cada autor selecionado. Temos o total de 24000 frases distintas por autor, totalizando em 72000 frases extraídas para o processo de treinamento e validação em cada um dos três conjuntos de dados. No processo de teste da rede são utilizadas 200 frases distintas por autor, totalizando 600 frases para cada conjunto de dados construído.

Tabela 4 – Quantidade total de sentenças extraídas por autor para o processo de treinamento, validação e teste da rede neural.

Processo	Autor	Qtd. Sentenças			Total
		D_1	D_2	D_3	
Treinamento/Validação	Faulkner	24000	24000	24000	72000
	Hemingway	24000	24000	24000	
	Roth	24000	24000	24000	
Teste	Faulkner	200	200	200	600
	Hemingway	200	200	200	
	Roth	200	200	200	

Fonte: Elaborada pelo autor (2021).

Note que a quantidade de sentenças extraídas para cada autor foi a mesma. Partimos da premissa de que o balanceamento dos conjuntos de dados é importante para evitar *bias* no processo de aprendizado da rede siamesa proposta (PETEEL, 2021).

3.3 PRÉ-PROCESSAMENTO TEXTUAL

A utilização das redes neurais para resolução de problemas complexos do cotidiano depende de processos que incluem inúmeras etapas a fim de obter resultados satisfatórios. Uma das etapas mais importantes é a de pré-processamento, por meio da qual elencamos quais dados são relevantes e que “fazem sentido” compor o conjunto de dados (COELHO, 2019).

O pré-processamento pode ser definido como o conjunto de atividades que englobam a preparação, a organização e a estruturação dos dados executadas antes da realização de análises e predições do modelo. Estima-se que esta etapa pode tomar até cerca de 80% do tempo e esforço em um projeto de análise de dados, além de influenciar diretamente a qualidade final resultados fornecidos pelos modelos (GOMES, 2019).

Em posse de versões disponibilizadas na internet das obras dos autores mencionadas na seção anterior, o processo em questão iniciou-se com a conversão das obras para arquivos no formato texto (.txt), os quais foram posteriormente lidos com a finalidade de se extrair as frases de cada autor conforme apresentado na Tabela 4.

A seleção de cada sentença passa por um processo de filtragem e seleção conforme apresentado no fluxograma da Figura 27, onde cada etapa do pré-processamento enumerada é detalhada logo abaixo dela. Para melhor visualização, a imagem disponibilizamos a imagem em nossa página no GitHub por meio do link <<https://github.com/HaraHeique/TCC-rede-neural-siamesa/blob/master/docs/images/pre-processamento-fluxograma-atualizado.png>>.

Figura 27 – Fluxograma do processo de filtragem e seleção das sentenças de textos extraídas das obras literárias.



Fonte: Elaborado pelo autor (2021).

1. **Remoção do conjunto de sentenças iniciais e finais:** primeiramente são removidas os conjuntos de textos iniciais e finais de cada obra, pois geralmente representam a capa, sobrecapa, sumário, considerações iniciais, finais e afins. Desta forma somente o conteúdo “em si” das obras foi levado em consideração para extração das sentenças;
2. **Verificação de sentença válida:** checa se a sentença de texto lida contém aspectos indesejados. Caso uma sentença seja considerada inválida, ela é ignorada e é lida a próxima sentença. Algumas das características que caracterizam sentenças como inválidas são: presença de marcações ou *flags* que as definem como títulos ou subtítulos de um capítulo, rodapés, enumerações de páginas e afins. Além disso, consideramos inválidas sentenças que possuem menos de 2 palavras ou mais de 150 palavras;
3. **Conversão para minúsculas:** realiza a transformação de todas as letras das palavras da sentença lida para minúsculas, no intuito de tanto auxiliar as etapas posteriores e uniformizar os dados, evitando problemas de *sensitive case*;
4. **Lematização das palavras:** esta é uma etapa relevante do processo que visa reduzir cada palavra da sentença lida ao seu radical chamado lema. Como resultado, obtém-se uma sentença em sua estrutura primitiva e sem terminações flexionadas. Entretanto, como é mostrado no fluxograma, esta etapa é somente executada caso o conjunto de dados corresponda ao D_3 (sem stopwords e lematizado);
5. **Remoção de pontuações:** basicamente remove todas as pontuações da sentença lida, tais como: pontos de interrogações, pontos de exclamações, pontos finais, hífens, vírgulas e afins, gerando uma nova sentença, pois consideramos que esses sinais gráficos não são relevantes para o resultado final do nosso modelo;
6. **Remoção de *tokens* não alfabéticos:** remove as palavras de uma sentença que não possuem todos os caracteres letras alfabéticas;
7. **Remoção de *stopwords*:** tem como objetivo remover as palavras consideradas “irrelevantes” na sentença. Dependendo do contexto, as *stopwords* podem ser facilmente ignoradas sem que a frase perca o seu significado. Todavia esta etapa, similarmente a etapa de lematização, só é executada caso o *conjunto de dados* seja o D_2 (sem stopwords) ou o D_3 (sem stopwords e lematizado);
8. **Reverificação de sentença válida:** executa novamente o mesmo conjunto de operações da primeira etapa, com o propósito de averiguar se a sentença lida e aplicada por todas as etapas anteriores de pré-processamento continua válida. Caso não seja mais válida, ela é ignorada e lê-se a próxima, iniciando novamente o ciclo a partir da primeira etapa. Caso contrário, adiciona-se a sentença pré-processada em uma estrutura de dados chamada lista, que é posteriormente usada para armazenar as frases em um arquivo CSV.

3.3.1 Técnicas de pré-processamento aplicadas

Com objetivo de ampliar o cenário investigativo desta pesquisa de TCC, optamos por estudar a influência de algumas técnicas de pré-processamento textual usuais na literatura do PLN, no intuito de analisar se o uso delas neste trabalho em questão influenciam no desempenho do modelo proposto na tarefa de determinação da similaridade entre os pares de sentenças dos autores. São elas: lematização das palavras, aplicada no *conjunto de dados D₃*, e remoção de *stopwords*, aplicada nos *conjuntos de dados D₂* e *D₃*.

A lematização é uma técnica de normalização que retorna a forma primitiva de uma palavra, conhecida como lema, a partir de suas diferentes formas flexionadas. Durante esse processo é realizada uma análise morfológica e de vocabulário da palavra (BALAKRISHNAN; LLOYD-YEMOH, 2014; BERGMANIS; GOLDWATER, 2018).

Existe outra técnica parecida com a lematização chamada *stemming*, a qual também reduz a palavra para uma forma mais básica por meio da remoção de sufixos, porém não leva em conta um análise morfológica da palavra. Supomos que para trabalhos voltados para a comparação de textos literários a lematização é um processo menos agressivo do que o *stemming*, o qual é uma opção mais simples, que é preferível quando o contexto das palavras não impacta significativamente nos resultados (PLISSON et al., 2004; KORENIUS et al., 2004; BALAKRISHNAN; LLOYD-YEMOH, 2014). A Tabela 5 demonstra as diferenças entre palavras da língua inglesa após a aplicação da técnica de lematização e do *stemming*.

Tabela 5 – Exemplos de palavras após a aplicação das técnicas de lematização e *stemming*, respectivamente.

Palavra	Lematização	Stemming
agree	agree	agre
with	with	with
policy	policy	polici
dictionary	dictionary	dictionari
are	be	ar
was	be	wa
were	be	were
speakers	speaker	speaker
studying	study	studi
separated	separate	separ
saving	save	save
saver	save	saver
saves	save	save

Fonte: Adaptada de (LUCCA; NUNES, 2002).

As *stopwords*, também conhecidas como palavras comuns, palavras irrelevantes ou componentes de um dicionário negativo, são palavras que, independente do idioma, aparecem

muitas vezes no texto, e que não agregam informação relevante ao texto (MAKREHCHI; KAMEL, 2008). Por possuírem baixo poder discriminativo e preditivo, podem, em várias aplicações, ser ignoradas com segurança sem que a sentença ou texto perca o seu significado, na medida que elas possuem apenas uma função sintática ou de ligação (EL-KHAIR, 2006; SAIF et al., 2014).

Como cerca de 30% a 50% das palavras em uma grande coleção de textos podem representar *stopwords*, existem vantagens ao se aplicar a técnica de remoção, tais como: i) a melhoria de desempenho em algumas tarefas que envolvem recuperação de informações e classificação de textos (EL-KHAIR, 2006); ii) a redução do conjunto de dados, impactando diretamente no tempo de treinamento de um modelo de RNA. Entretanto, um ponto que se deve atentar é algumas palavras tidas como “irrelevantes” pela literatura de PLN podem mudar o contexto da frase ou texto, provocando problemas de desempenho na tarefa em que está sendo aplicada (TEJA, 2020). A Tabela 6 fornece exemplos de frases da língua inglesa que passam pelo processo de remoção de *stopwords*. Tal remoção se deu com a utilização da biblioteca *NLTK* (*Natural Language Toolkit*), versão 3.2.5, com 174 *tokens* padrão. O lado esquerdo da tabela contém as frases com *stopwords* (em negrito) e o lado direito as mesmas frases sem as *stopwords*.

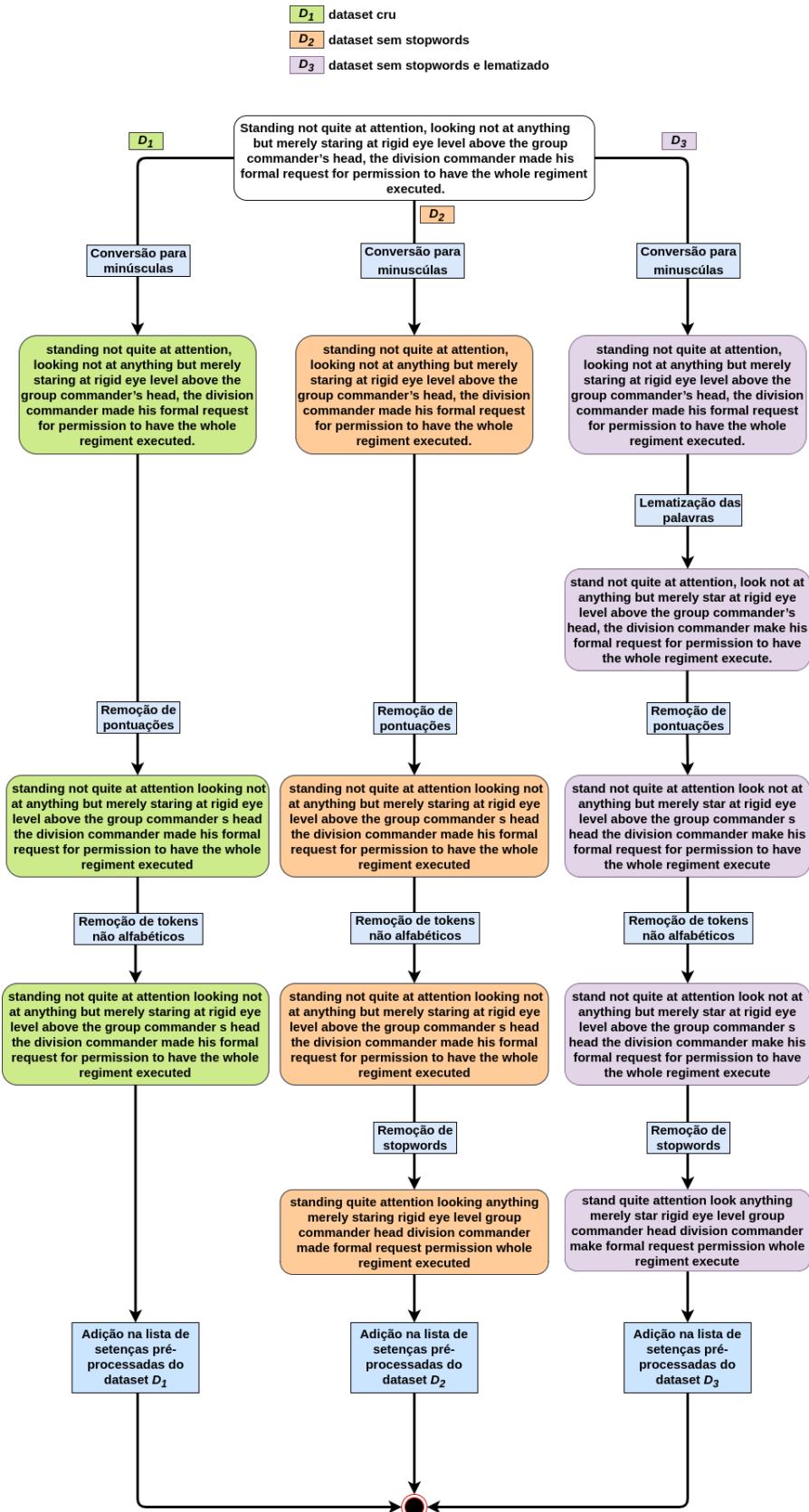
Tabela 6 – Frases com stopwords × Frases sem stopwords.

Frase com stopwords	Frase sem stopwords
This is a sample sentence, showing off the stop words filtration.	sample sentence, showing stop words filtration.
When you don't take any risks, you risk everything.	take risks, risk everything.
Are necessary too many years of work to succeed overnight.	necessary many years work succeed overnight.
Be the change you want to see in the world.	change want see world.
If you set goals ridiculously high and it's a failure, you will fail above everyone else's success.	set goals ridiculously high failure, fail everyone else success.

Fonte: Elaborada pelo autor (2021).

A aplicação de técnicas de pré-processamento apresentadas na Figura 27, antes do treinamento do modelo de aprendizado profundo proposto no trabalho foi realizada com o intuito de se criar um amplo cenário investigativo. A Figura 28, apresentada abaixo, sumariza toda a transformação de uma sentença específica de texto até que ela faça parte de algum dos conjuntos de dados D_1 , D_2 ou D_3 . Para melhor visualização por parte do leitor, a imagem pode ser acessada em nosso GitHub por meio do link <<https://github.com/HaraHeique/TCC-rede-neural-siamesa/blob/master/docs/images/pre-processamento-exemplo-sentenca.png>>.

Figura 28 – Exemplo de pré-processamento de uma sentença válida extraída da obra *A Fable* (1954), do autor Faulkner, seguindo as etapas do fluxograma da Figura 27.



Fonte: Elaborado pelo autor (2021).

Todo o processo de pré-processamento apresentado nesta seção foi executado em dois momentos. Primeiramente, na geração dos três conjuntos de dados (D_1 , D_2 e D_3) **de treinamento e validação**, e, em segundo lugar, na geração dos três conjuntos de dados (D_1 , D_2 e D_3) **de teste**.

3.4 ESTRUTURAÇÃO DO CONJUNTO DE DADOS

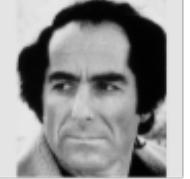
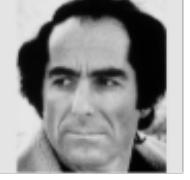
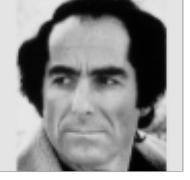
Quando se está trabalhando com aplicações que usam aprendizado de máquina, são muito comuns tarefas que envolvem análise e captura de dados nos quais se deseja encontrar correlações e identificar padrões (RENAUX, 2018; MOLNAR, 2020). Geralmente os dados passam por um pré-processamento e em um dado momento são armazenados para serem posteriormente utilizados em outras etapas (TOPRAK, 2019).

Uma das formas muito utilizadas para o armazenamento do conjunto de dados são arquivos do formato *Comma-Separated Values*, comumente conhecidos como arquivos CSV. Esses tipos de arquivos são caracterizados pela sua forma compacta de representar dados em tabelas, onde cada vírgula separa os dados das colunas. São amplamente utilizados devido às vantagens que apresentam em relação a outros tipos de representação: são suportados por diferentes softwares, têm estrutura simples e são fáceis de manipular e de armazenar (TEJADA et al., 2018).

Neste trabalho foram utilizados arquivos CSV para armazenamento dos conjuntos de dados pré-processados de treinamento, validação e teste.

Um ponto vital para que uma RNA extraia features e aprenda a partir deles reside na coleta adequada dos resultados e na estruturação correta dos conjuntos de treinamento, validação e teste. Posto isto, os conjuntos de dados de treinamento e validação foram organizados na forma de combinações de sentenças entre todos os pares possíveis de autores, como é mostrado no esquema da Figura 29. Note que temos 12000 sentenças de texto combinadas em pares para cada par de autores, sendo 6000 sentenças provenientes de obras do autor 1 (lado esquerdo) e as outras 6000 sentenças do autor 2 (lado direito).

Figura 29 – Esquema da estruturação das sentenças dos conjuntos de dados de treinamento e validação.

Sentenças de treinamento	Autor 1	Autor 2
1 a 12000	 Faulkner	 Faulkner
12001 a 24000	 Hemingway	 Hemingway
24001 a 36000	 Roth	 Roth
36001 a 48000	 Faulkner	 Hemingway
48001 a 60000	 Faulkner	 Roth
60001 a 72000	 Hemingway	 Roth

Fonte: Elaborado pelo autor (2021).

A Tabela 7 mostra como são estruturados os arquivos CSV dos três conjuntos de dados (D_1 , D_2 e D_3) os quais estão dispostos em 7 colunas, as quais são: *phrase1* e *phrase2*, que são as sentenças pré-processadas nas etapas de filtragem do fluxograma da Figura 27, *qd1* e *qd2*, que são os identificadores (IDs) das colunas *phrase1* e *phrase2* respectivamente, a coluna *label*, que representa o índice de similaridade entre as sentenças das colunas *phrase1* e *phrase2*, sendo que o valor do índice é sempre 0, indicando sentenças de diferentes autores, e 1, indicando sentenças de mesmos autores, e, por fim, as colunas *author1* e *author2*, que determinam quais são os autores da *phrase1* e da *phrase2*, respectivamente. Como os conjuntos de dados possuem duas colunas de sentenças e 36000 linhas correspondente aos dados em si tem-se o total de 72000 sentenças de texto para cada conjunto de dados.

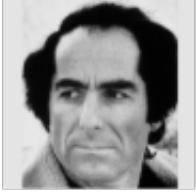
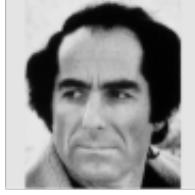
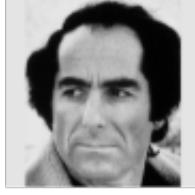
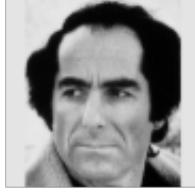
Tabela 7 – Exemplos de sentenças do arquivo CSV do conjunto de dados D_3 de treinamento e validação.

qd1	qd2	phrase1	phrase2	label	author1	author2
11	12	nothing whatever group commander face	endorsed receive say	1	faulkner	faulkner
13567	13568	thought division manpower	squabble division kill get	1	hemingway	hemingway
25903	25904	harry master stood directly beside swede indifferent bos word work	fortyone year newark maid work	1	roth	roth
43379	43380	pick though newcomer say	like start write	0	faulkner	hemingway
55525	55526	sunburn maybe ingrain dirt turn white	thing people discard roseanna collect	0	faulkner	roth
68761	68762	sergeant lay dirty longsleeved underwear	see anyone father damrosch	0	hemingway	roth

Fonte: Elaborado pelo autor (2021).

Quanto aos conjuntos de dados de teste, optamos pela seguinte abordagem: extraímos novas sentenças de texto de obras não utilizadas para construção dos conjuntos de treinamento e validação e construímos todas as combinações de sentenças entre todos os pares possíveis de autores, conforme ilustrado na Figura 30. Note que são 100 sentenças de texto combinadas em pares para cada par de autores, sendo 50 sentenças provenientes da obra do autor 1 (lado esquerdo) e as outras 50 sentenças do autor 2 (lado direito). No total, há 600 sentenças.

Figura 30 – Esquema da estruturação das sentenças dos conjuntos de dados de teste.

Sentenças de teste	Autor 1	Autor 2
1 a 100	 Faulkner	 Faulkner
101 a 200	 Hemingway	 Hemingway
201 a 300	 Roth	 Roth
301 a 400	 Faulkner	 Hemingway
401 a 500	 Faulkner	 Roth
501 a 600	 Hemingway	 Roth

Fonte: Elaborado pelo autor (2021).

A Tabela 8 apresenta os arquivos CSV dos três conjuntos de dados (D_1 , D_2 e D_3) de teste, que são dispostos de modo similar aos de treinamento e validação. As colunas presentes são: *phrase1* e *phrase2*, que contêm as sentenças extraídas e já pré-processadas; *author1* e *author2*, que identificam os autores das colunas *phrase1* e *phrase2*, respectivamente; e a coluna *label*, determinando o índice de similaridade entre as sentenças das colunas *phrase1* e *phrase2*, onde sentenças de mesmos autores são definidas como 1 e de autores diferentes definidas como 0.

Tabela 8 – Exemplos de sentenças retiradas do arquivo CSV do conjunto de dados D_3 (sem stopwords e lematizado) de teste.

phrase1	phrase2	author1	author2	label
father quentin cant hurt	mother put handkerchief veil	faulkner	faulkner	1
reel line first told	hooked fish lose without wake eddy	hemingway	hemingway	1
strong character big bankroll evidence one worth	good sensible people	roth	roth	1
saw eye ran hill	know seem like lot money lot money bought tackle	faulkner	hemingway	0
bet right tent open	two offspring waste child wed disaster difficult understand	faulkner	roth	0
one table play domino already	bear lose little punch bag	hemingway	roth	0

Fonte: Elaborada pelo autor (2021).

Após a estruturação de todos os seis conjuntos de dados, três para treinamento e validação e três para testes, é realizado o seu armazenamento dentro da própria aplicação desenvolvida, não havendo mais necessidade de efetuar todo o processo de captura, análise e filtragem dos dados toda vez que se realizar treinamentos, validações ou testes na RNA, bastando recuperá-los usando os conjuntos de dados devidamente estruturados.

3.4.1 Word Embedding Utilizado

Uma abordagem fundamental utilizada no trabalho para que a rede neural siamesa capture significados de uma palavra em um documento ou texto a fim de aprender a determinar a similaridade entre os pares de sentenças literárias é chamada *word embedding*. Ela é considerada uma das principais técnicas de vetorização utilizadas em problemas que envolvem resolução de tarefas complexas da área do PLN (KARANI, 2018; BROWNLEE, 2019a).

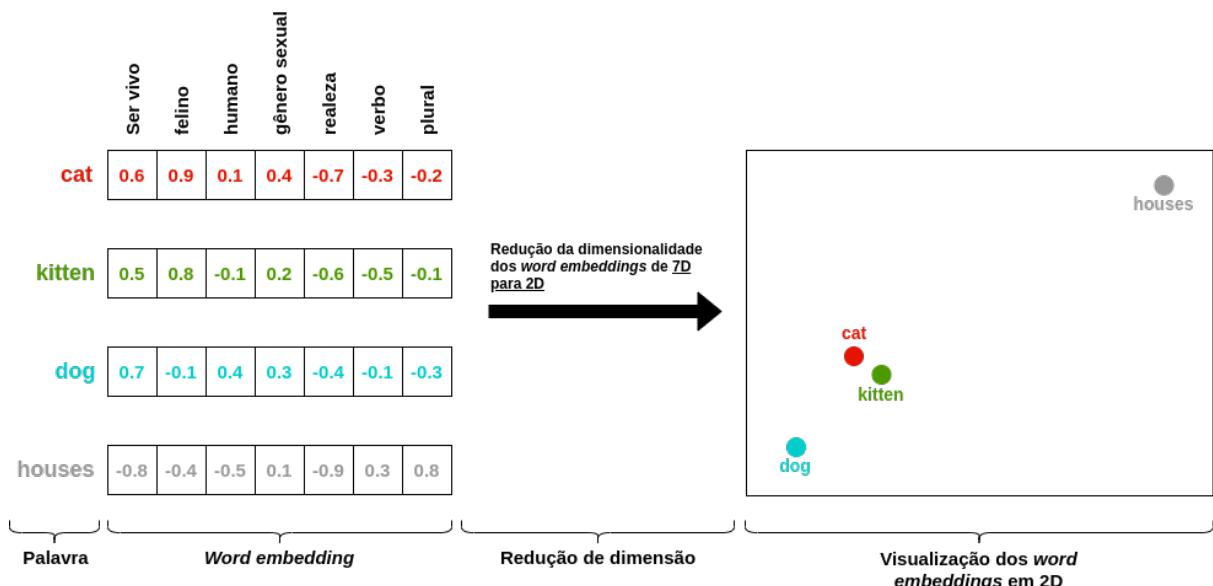
Segundo ALMEIDA e XEXÉO (2019), *word embedding* é uma abordagem de mineração de textos que transforma palavras em vetores densos multidimensionais de comprimento fixo, que são criados usando estatísticas de co-ocorrência de palavras ou de contexto (ALMEIDA; XEXÉO, 2019). Esses vetores são compostos por números reais que indicam diferentes *features* das palavras. Cada uma das palavras é representada por um ponto

em um espaço multidimensional, normalmente conhecido como *embedding space*. Nesse espaço, quanto mais próxima é a representação (posição) de duas palavras mais similar é o seu significado/sentido (MOURA, 2018; BROWNLEE, 2019a).

Duas amplas classes de *word embedding* incluem as baseadas em frequências de palavras e as baseadas em modelos que utilizam dicionários léxicos (PANETTO et al., 2019). Neste trabalho foi aplicada a técnica *Word2Vec* para obtenção dos vetores densos de representação de cada palavra (MIKOLOV et al., 2013).

O *Word2Vec* é um método estatístico que aprende de forma eficiente a representar uma palavra por um vetor denso usando uma RNA rasa. O *Word2Vec* pode ser configurado usando a estratégia *Continous Bag of Words (CBOW)* ou a estratégia *Skip Gram* (BROWNLEE, 2019a). A entrada da rede é alimentada por um corpus de texto normalmente amplo, que é processado pelo modelo com o intuito de extrair *features* das palavras e aprender com elas. A saída é um vocabulário com as palavras e seus vetores, onde cada vetor é a localização da palavra num espaço vetorial multidimensional. A Figura 31 ilustra a representação de quatro palavras pelo *Word2Vec* em vetores de 7 dimensões e sua a representação geométrica em um espaço bidimensional. Note que as palavras “*cat*” e “*kitten*” estão próximas entre si por possuírem o mesmo significado, enquanto que a palavra “*houses*” encontra-se distante das outras palavras por ter significado semântico e sintático diferente (TOMALOK, 2018).

Figura 31 – Processo de operação do *Word2Vec*.

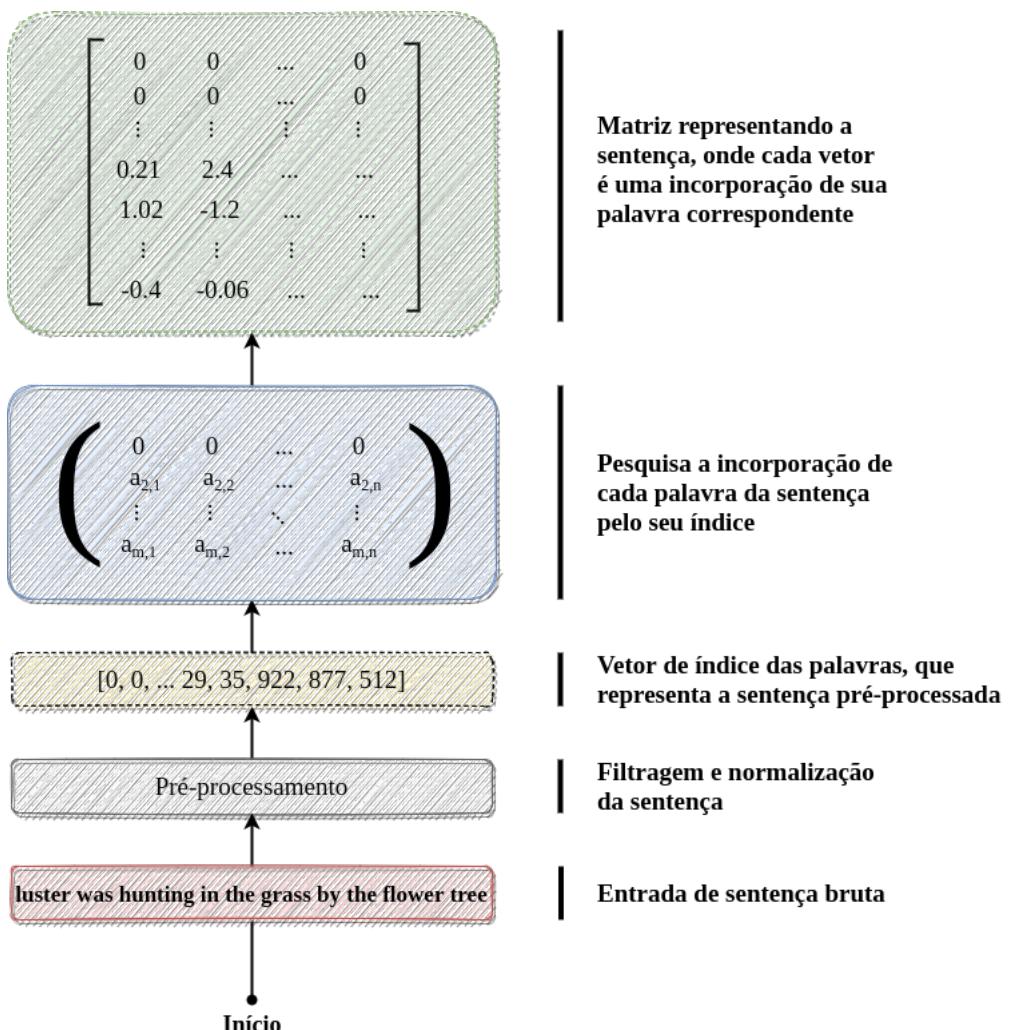


Fonte: Imagem adaptada de <<https://ichi.pro/pt/treinar-word2vec-usando-gensim-22278944385252>>. Acesso em: 17 de jun. 2021.

Neste TCC, a aplicação do *Word2Vec* ocorre antes de alimentar a rede com os dados, pois ele é responsável por criar a representação das palavras. No nosso caso são utilizados vetores de 300 dimensões para cada palavra extraída das bases de dados.

A Figura 32 apresenta um exemplo de como funciona o Word2Vec neste TCC. Uma sentença de texto bruta passa primeiramente pelo pré-processamento, apresentado pelo fluxograma da Figura 27. Após isto, é criado um vetor de comprimento fixo que contém os índices das palavras contidas na sentença, em que os primeiros zeros correspondem ao *padding* à esquerda que são ignorados, porém são necessários para que todas as sentenças contenham o mesmo tamanho no vetor, enquanto que os valores diferentes de zero, presentes sempre à direita, correspondem aos índices que identificam as palavras da sentença de maneira exclusiva. Este vetor serve como entrada de uma das subredes da rede siamesa passando pela camada de incorporação, que é responsável por encontrar o *embedding* correspondente para cada palavra da sentença representada pelo vetor de índices e encapsulá-los em uma matriz, que representa a sentença de texto bruta como uma sequência de números reais. Foi utilizado o modelo de Word2Vec pré-treinado da Google que pode ser baixado em <<https://code.google.com/archive/p/word2vec/>>.

Figura 32 – Exemplo de um processo de *word embedding* de uma sentença de texto de um dos conjuntos de dados.



Fonte: Adaptada de (COHEN, 2017).

Existem duas formas de aplicarmos o `Word2Vec` para criação de representações densas de palavras: treinando o próprio modelo usando algoritmos mencionados anteriormente ou utilizando um modelo pré-treinado. Neste trabalho foi utilizado um modelo pré-treinado de `Word2Vec` da *Google*. Este modelo escolhido foi inicialmente treinado por pesquisadores da *Google* utilizando dados provenientes do *Google News* com o tamanho do *corpus* textual contendo em torno de 100 bilhões de *tokens* (palavras) em língua inglesa. É caracterizado por conter vetores de 300 dimensões.

3.5 DEFINIÇÃO DE PARÂMETROS E HIPERPARÂMETROS

Cientistas de dados e pesquisadores da área de aprendizado de máquina buscam construir um modelo de aprendizado que seja capaz de aprender a extrair informação de dados e adquirir uma capacidade de generalização que permita a resolução de um problema específico de forma eficaz. Porém, para alcançar isto é necessário encontrar um bom conjunto de configurações para o modelo com o qual se está trabalhando (GOZZOLI, 2018).

Essas configurações definidas antes e até durante o treinamento de algoritmos de aprendizado profundo são realizadas nos hiperparâmetros. Eles são variáveis que determinam a estrutura e arquitetura do modelo de RNA, assim como a maneira como ela deve se comportar (RADHAKRISHNAN, 2017). É fundamental realizar pesquisas acerca hiperparâmetros a fim de encontrar o conjunto “ideal” de combinações para diferentes conjuntos de dados baseando-se no contexto de resolução do problema (ALRUBAN, 2019).

Além dos hiperparâmetros dos modelos de RNA também existem os parâmetros dos modelos. Ambos são frequentemente usados de maneira intercambiável, gerando assim confusões pela inconsistência no uso desses termos. BROWNLEE (2019) diz que um parâmetro é uma variável de configuração interna ao modelo que pode ser exclusivamente estimada e aprendida através dos dados, enquanto que um hiperparâmetro não é estimado a partir dos dados e são frequentemente ajustados conforme o contexto do problema de modelagem preditiva, podendo esses ajustes serem feitos de forma manual ou utilizando heurísticas (BROWNLEE, 2019b).

3.5.1 Definição do Tamanho Máximo de Sequência

Como mencionado nas seções anteriores, este TCC envolve três conjuntos de dados. Para cada um deles foi plotado um histograma para visualizarmos geometricamente a distribuição estatística das sentenças a fim de determinar o conjunto de escolhas do tamanho máximo de sequência, denominada aqui como *max_seq_length*.

O tamanho máximo de sequência define qual o tamanho fixo de todas as sentenças de texto de um conjunto de dados. Independentemente do valor dessa variável, todas

as sentenças de um conjunto de dados serão representadas por matrizes de dimensões $\text{max_seq_length} \times 300$.

A partir de testes empíricos, elencamos três possibilidades de valor para a variável max_seq_length . Tomamos como base medidas estatísticas de tendência central obtidas das distribuições de sentenças dos histogramas plotados para cada conjunto de dados. Os histogramas estão mostrados na Figura 33.

As medidas de tendência central escolhidas foram: a média aritmética μ (número médio de palavras das sentenças), a mediana η (número de palavras tal que metade das sentenças do conjunto de dados possui quantidade de palavras maior do que η) e a constante $\gamma = \frac{\mu + \eta}{2}$.

O principal motivo dessas escolhas foi porque há uma grande concentração da distribuição do tamanho das sentenças em torno dessas medidas. Abaixo é apresentada a Tabela 9 com os valores de tamanho máximo de sequência escolhidos para cada conjunto de dados utilizado neste TCC. Note ainda que estudar a influência da troca de valores para a variável max_seq_length também amplia o cenário de investigação atrelado a este TCC.

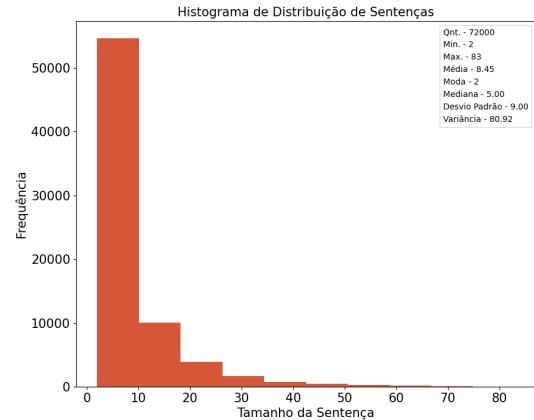
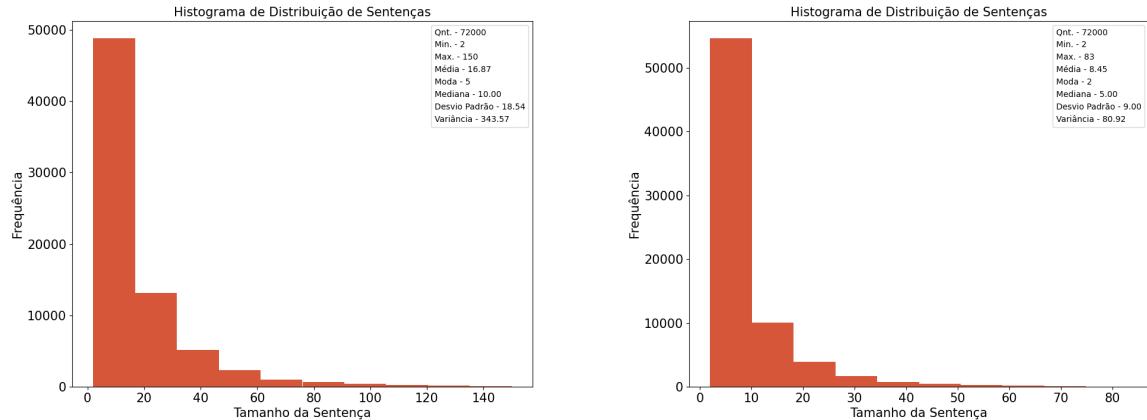
Tabela 9 – Tamanhos máximos das sequências escolhidas para cada conjunto de dados.

Conjunto de dados	tamanho máximo de sequência (max_seq_length)
D_1	$\mu = 17, \eta = 10, \gamma = 14$
D_2	$\mu = 9, \eta = 5, \gamma = 7$
D_3	$\mu = 9, \eta = 5, \gamma = 7$

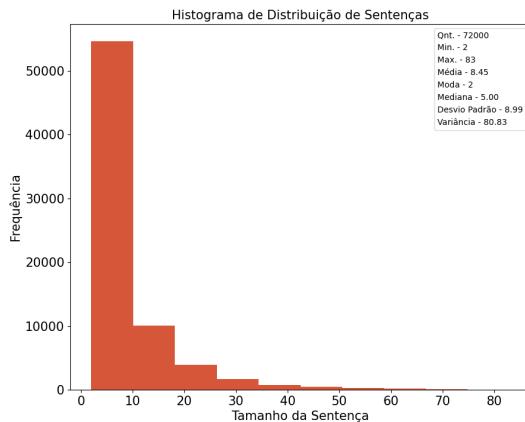
Fonte: Elaborada pelo autor (2021).

Figura 33 – Histogramas da quantidade de sentenças em função da quantidade de palavras de cada sentença.

(a) Conjunto de dados D_1 (sem pré-processamento) (b) Conjunto de dados D_2 (sem stopwords)



(c) Conjunto de dados D_3 (sem stopwords e lematizado)



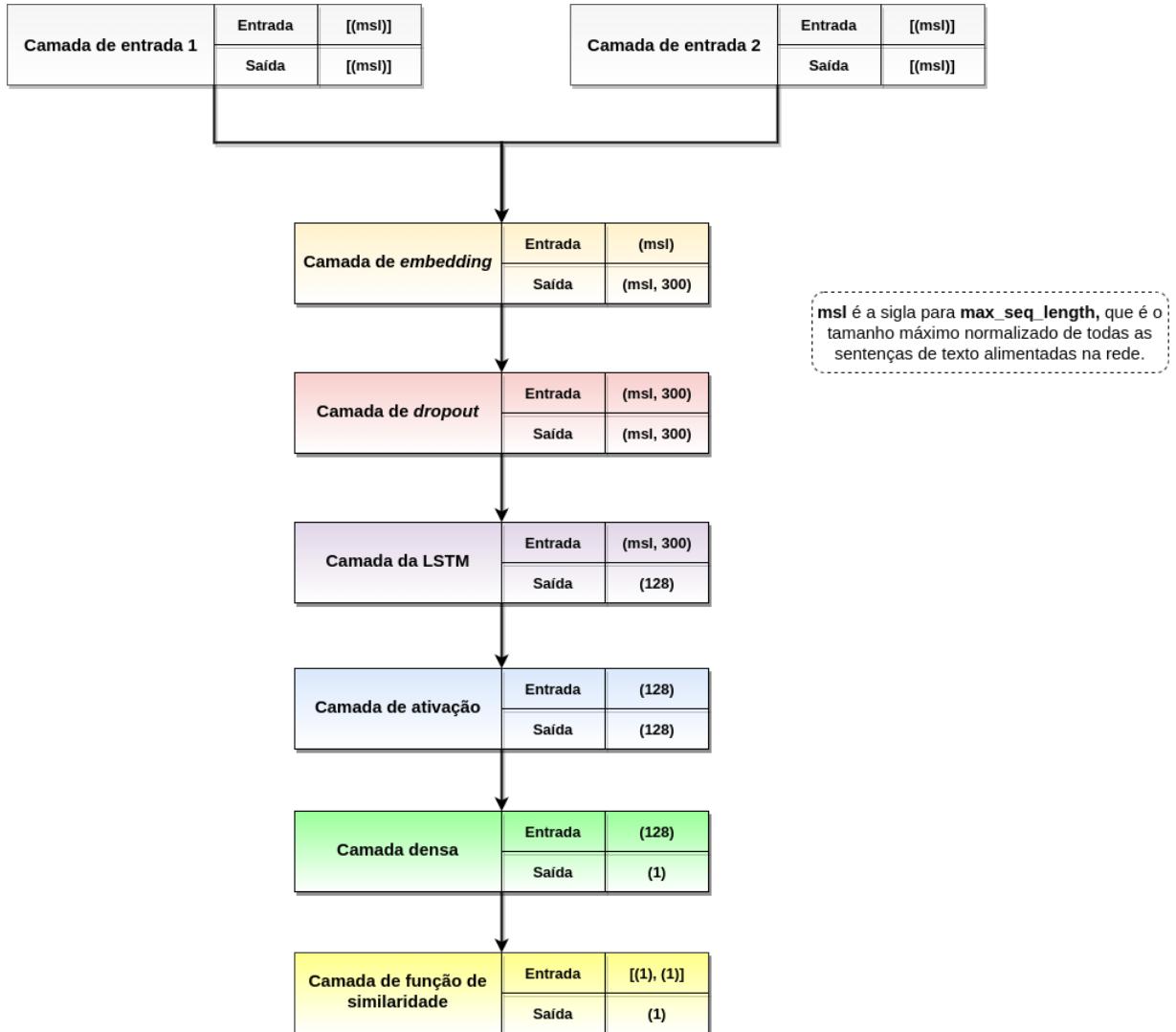
Fonte: Elaborado pelo autor (2021).

3.5.2 Arquitetura da Rede Siamesa

A Figura 34 apresenta a arquitetura da rede neural siamesa usada no trabalho. Note que ela é composta por sete camadas principais, as quais são: camadas de entrada, camada de *embedding*, camada de *dropout*, camada da LSTM, camada de ativação, camada densa e, por último, a camada de função de similaridade.

A primeira camada é a de entrada, e como o próprio nome sugere, é nela que os vetores de índices referentes a matriz de *embedding* são introduzidos. Note que na Figura 34 existem duas camadas de entrada, pois como se trata de uma rede siamesa são fornecidos pares de entrada e cada camada de entrada se remete a uma de suas sub-redes.

Figura 34 – Arquitetura do modelo da rede neural siamesa.



Fonte: Elaborada pelo autor (2021).

Na camada de *embedding* cada índice dos vetores de entrada é convertido em seu vetor de números reais de 300 dimensões. Os detalhes acerca deste processo aplicado nesta camada foram explicados na subseção 3.4.1 na Figura 32.

A camada de *dropout* é responsável por aplicar o método de regularização *dropout* à entrada de rede. Nela, os componentes de entrada são zerados aleatoriamente com uma dada taxa com objetivo de evitar que ocorra o problema de *overfitting* da RNA, e de melhorar a sua performance (SRIVASTAVA, 2013; SRIVASTAVA et al., 2014).

Na camada da LSTM extraídas as *features* dos textos e é onde a rede aprende a generalizar a partir do processo de aprendizado. Ela é a estrutura principal que pode ser substituída por outras estruturas de rede a fim de encontrar o modelo de aprendizagem com o melhor desempenho no problema que se pretende resolver. Foram realizados tanto testes empíricos quanto testes usando uma heurística Bayesiana em busca de um conjunto de parâmetros e

hiperparâmetros que levasse a LSTM a melhorar a sua performance diante do problema proposto. Os valores definidos para esses parâmetros e hiperparâmetros serão detalhados na próxima seção.

A saída da LSTM passa por uma camada de ativação, onde é aplicada uma função de ativação capaz de decidir se as conexões externas devem considerar se os neurônios artificiais estão ativos ou inativos, influenciando na capacidade de generalização e no desempenho do modelo (SHARMA, 2017). No trabalho foi usada a função de ativação *Scaled Exponential Linear Unit* (SELU) (MARCHISIO et al., 2018). A SELU é capaz de se auto-normalizar, o que a torna sua aplicação mais rápida do que as outras funções de ativação da família RELU, além de evitar os problemas de desaparecimento e explosão de gradiente (BÖHM, 2018; HANSEN, 2019).

Na camada densa ocorre a alteração da dimensionalidade dos vetores de características fornecido pelas camadas anteriores (SHARMA, 2020b). No trabalho é realizada a conversão da dimensão do espaço de saída de dimensão 128 para dimensão 1 e logo em seguida aplicada à uma função de ativação sigmóide, gerando uma saída numérica real no intervalo fechado entre 0 e 1.

Na camada de função de similaridade, a rede siamesa aprende a diferenciar os pares de entrada de uma mesma classe ou de classes diferentes a partir da aplicação da medida de similaridade de Manhattan.

3.5.3 Hiperparâmetros definidos usando uma heurística Bayesiana

Escolher o conjunto “ideal” de hiperparâmetros para o modelo de aprendizado de acordo com o contexto do problema é uma tarefa crucial na aprendizagem de máquina, que pode ser árdua, cansativa e demorada de se realizar (BISSUEL, 2019). Uma das abordagens utilizada para encontrar o conjunto de hiperparâmetros que resulte em um modelo que forneça o melhor desempenho medido em um conjunto de dados de validação é chamado de otimização de hiperparâmetros ou ajuste de hiperparâmetros (BROWNLEE, 2020b; CHAUHAN, 2020).

O processo de otimização de hiperparâmetros ocorre através da aplicação de diferentes algoritmos ou técnicas que buscam encontrar os hiperparâmetros candidatos a melhorar a acurácia do modelo de aprendizagem (VAZ, 2019). Algumas delas são: ajuste manual, pesquisa em grade, pesquisa aleatória e por fim a aplicada neste trabalho: a otimização Bayesiana.

A otimização Bayesiana vem ganhando popularidade nos últimos anos na tarefa de escolher os melhores hiperparâmetros para os modelos de aprendizado (SHAHRIARI et al., 2016). O seu diferencial em relação aos outros algoritmos é a sua capacidade de escolher os

hiperparâmetros candidatos a serem mais promissores com base em resultados de avaliações anteriores, ou seja sem realizar uma busca exaustiva (WU et al., 2019; ARASANIPALAI, 2021). A otimização Bayesiana usa probabilidades condicionais e o teorema de Bayes para determinar scores para as configurações mais prováveis de fazer com a rede performe melhor. Ou seja, o algoritmo de otimização Bayesiana cria um ranking das configurações mais recomendadas ao modelo (FRAZIER, 2018; CHAUHAN, 2020).

Neste TCC foi aplicada a heurística Bayesiana para determinação dos hiperparâmetros usando-se a ferramenta *Hyperas*². Ela cria uma espécie de invólucro em torno da biblioteca *Python Hyperopt*³ dando a capacidade de utilizar as funcionalidades do *Hyperopt* sem aprender a sua sintaxe, utilizando-a de uma maneira mais fácil e direta para descobrir os melhores conjuntos de hiperparâmetros para o modelo.

A Tabela 10 contém todos os hiperparâmetros elencados por nós empiricamente e os que escolhidos pelo *Hyperas*, que estão destacados em negrito.

Tabela 10 – Hiperparâmetros do modelo de aprendizagem testados na ferramenta *Hyperas*.

Hiperparâmetros	Conjunto de valores testados
Units	64, 128 , 256
Activation	Softsign , Tanh
Recurrent Activation	Sigmoid, Hard Sigmoid
Dropout Layer	[0, ..., 0.2469 , ..., 1]
Dropout	[0, ..., 0.8228 , ..., 1]
Recurrent Dropout	[0, ..., 0.0653 , ..., 1]
Kernel Initializer	Variance Scaling, Glorot Normal
Batch Size	16, 32, 64 , 128, 256
Optimizer	Adadelta , Adamax, Adagrad, SGD, RMSProp
Learning Rate	0.001, 0.01, 0.1
Clipnorm	0, 0.75, 1.5 , 2, 3
Loss	BinaryCrossentropy, MeanSquaredError , ConstrativeLoss
Activation Layer	Selu , Relu, Elu
Activation Dense Layer	Sigmoid , Hard Sigmoid

Fonte: Elaborado pelo autor (2021).

3.6 RECURSOS COMPUTACIONAIS UTILIZADOS

Neste trabalho foram utilizados tanto recursos de *software* quanto de *hardware* para execução dos experimentos de treinamento, validação e teste da rede neural siamesa.

Em relação aos recursos de *software*, foi utilizada a linguagem de programação *Python* versão 3.X, pelo fato de ela ser uma linguagem de *script* de fácil manuseio e de baixa

² <<https://github.com/maxpumperla/hyperas>>. Acesso em: 09 set. 2021

³ <<https://github.com/hyperopt/hyperopt>>. Acesso em: 09 set. 2021

curva de aprendizagem, além de ser altamente utilizada na área de ciência de dados e em implementações de algoritmos de *machine learning*, contendo diversos *frameworks*, *toolkits* e bibliotecas de suporte. Dentre os principais *frameworks* e bibliotecas utilizadas no trabalho para implementação da rede neural siamesa que envolvem tarefas de processamento de linguagem natural estão: *Tensorflow*, *Keras*, *NumPy*, *pandas*, *NLTK* e *Matplotlib*. Todas estas dependências citadas, assim como o código fonte escrito em *Python* e os resultados obtidos estão localizados no repositório remoto deste trabalho, no Github⁴. A implementação da rede siamesa que foi adaptada e utilizada no trabalho, conhecida como MaLSTM⁵, também se encontra hospedada na mesma plataforma.

Quanto aos recursos de *hardware*, foram utilizados nos experimentos dois computadores com configurações suficientemente para o conjunto de testes realizados no trabalho.

Tabela 11 – Configurações dos computadores utilizados para execução dos experimentos.

Máquina	Configurações
Computador 1	CPU: Intel Core i9-9900KF (3.6 GHz, 8 núcleos, 16 threads) GPU: Nvidia GeForce RTX TM 2060 (6GB GDDR6, 1.680 GHz) RAM: 32GB (DDR4, 2.666 GHz)
Computador 2	CPU: Intel Core i7-8750H (2.20 GHz, 6 núcleos, 12 threads) GPU: Nvidia GeForce GTX 1060 Max-Q (6GB GDDR6, 1.265 GHz) RAM: 16GB (DDR4, 2.666 GHz)

Fonte: Elaborado pelo autor (2021).

Durante os experimentos com ambos computadores, apresentados na Tabela 11, foi possível notar que especificamente no processo de carregamento do *Word2Vec*, o consumo de memória RAM sobe em torno de 5GB, sendo necessário que a máquina tenha pelo menos 6GB de RAM disponíveis durante a execução do algoritmo, sem que o Sistema Operacional precise realizar *swapping*, ou até mesmo travar outros processos triviais por insuficiência de memória. Outro ponto notado foi que o fato de ambas as placas de vídeo dos computadores possuirem suporte a tecnologia CUDA agilizou notavelmente o tempo de treinamento da rede. Tal fato não ocorreria caso fosse utilizado a CPU, especialmente com o aumento de épocas no treinamento do modelo.

Também foram efetuados testes pontuais no *Google Colab*, que disponibiliza o uso de GPUs para agilizar o treinamento de RNAs. Entretanto, não foi o meio preferível, devido as limitações de recursos empregadas pela plataforma, que em alguns casos dificultava ou até mesmo impedia a execução do trabalho por um período de tempo.

⁴ <<https://github.com/HaraHeique/TCC-rede-neural-siamesa>>. Acesso em: 20 ago. 2021

⁵ <<https://github.com/likejazz/Siamese-LSTM>>. Acesso em: 08 jun. 2021

3.7 SOBRE A REPRODUTIBILIDADE DESTE TRABALHO

A arquitetura da aplicação desenvolvida é baseada em responsabilidades, onde cada pacote com seus respectivos módulos presentes no código fonte possuem um conjunto de funções bem definidas a fim de dividir as etapas de captura e pré-processamento dos dados, treinamento, validação e testes da RNA proposta. O código fonte, as descrições gerais dos principais módulos e arquivos contidos na aplicação e detalhes de sua execução estão disponibilizados no Github no link <<https://github.com/HaraHeique/TCC-rede-neural-siamesa>>.

Além disso, para fins de reproduzibilidade deste trabalho todas as bases de dados construídas estão disponíveis no repositório Mendeley Data sob licença *Creative Commons Attribution 4.0*, no link <<https://data.mendeley.com/datasets/tg6pxsnxr5/1>>, contendo descrições e explicações de como utilizá-la.

4 ANÁLISES DE RESULTADOS

Neste capítulo são apresentados os resultados dos experimentos computacionais realizados acerca da análise de performance da rede siamesa na tarefa de obtenção da similaridade de Manhattan entre pares de sentenças literárias.

A rede siamesa aplicada no trabalho foi baseada na implementação da MaLSTM proveniente do artigo *How to predict Quora Question Pairs using Siamese Manhattan LSTM*, que foi utilizada para resolver o problema do desafio Kaggle intitulado de *Quora Question Pairs*⁸ (COHEN, 2017). O desafio *Quora Question Pairs* foi uma inspiração para este TCC e seu principal objetivo era identificar pares de perguntas com a mesma intenção e significado através de ferramentas avançadas da área de PLN capazes de verificar se as questões eram duplicadas ou não.

Todavia, a abordagem empregada no desafio é de contexto mais simples do que a aplicada neste TCC, pois é de caráter mais sintático e menos semântico, dado que essencialmente a ideia do desafio do Kaggle é determinar se dois trechos de textos estão expressando a mesma coisa, ou seja, se estão “duplicados”.

Em contrapartida, determinar a similaridade entre dois trechos de textos, que são sentenças literárias extraídas das obras de autores, depende fortemente de estruturas sintáticas, semânticas, nuances pessoais e subjetividades, características naturais presentes em obras literárias. A Figura 35 compara os pares de frases do desafio *Kaggle* com pares de frases das bases de dados deste TCC.

Figura 35 – Comparaçõa de frases retiradas do desafio *Kaggle* com as deste TCC.



Desafio Kaggle					
id	qid1	qid2	question1	question2	is_duplicate
7	15	16	How can I be a good geologist?	What should I do to be a great geologist?	1
22	45	46	What are the questions should not ask on Quora?	Which question should I ask on Quora?	0

Textos Literários (TCC)						
qd1	qd2	phrase1	phrase2	label	author1	author2
19	20	the division commander had not moved	the group commander looked at him	1	faulkner	faulkner
69999	70000	i had expected it to be flatter more like a plateau	you kissed him back	0	hemingway	roth

Fonte: Elaborado pelo autor (2021).

⁸ <<https://www.kaggle.com/c/quora-question-pairs>>. Acesso em: 09 set. 2021

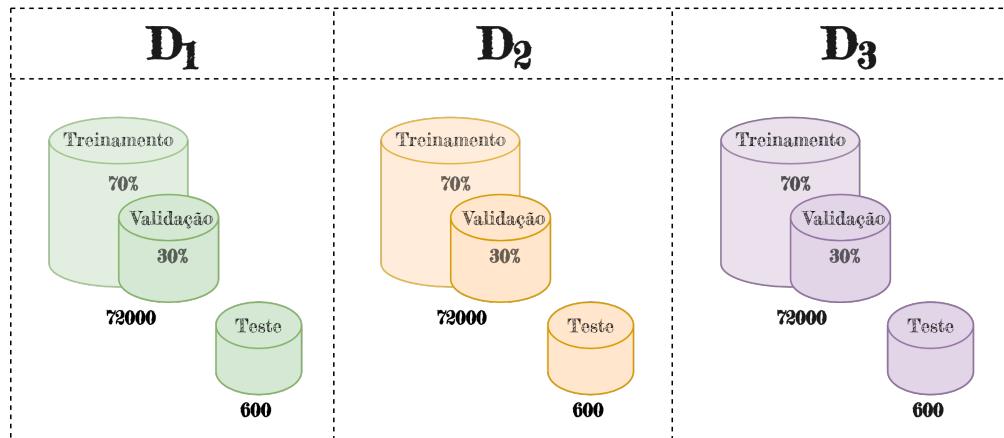
Nesse sentido, este TCC possui um caráter desafiador e complexo, dado que a ideia central é verificar se as redes siamesas LSTM conseguem capturar níveis de similaridade mais complexas e subjetivas, a fim de capturar os *ethos* artístico-literários característicos dos autores escolhidos e, assim, identificar por meio da medida de similaridade de Manhattan se as sentenças literárias tendem ou não a serem de mesmos autores.

Devido à natureza complexa do problema, os resultados obtidos que serão apresentados adiante nesta seção não foram tão promissores quanto do modelo utilizado no desafio *Kaggle*, mas foi realizada uma ampla investigação científica neste trabalho que aponta caminhos e rotas e contribui para diversos trabalhos futuros, na nossa opinião.

A fim de analisar o desempenho da rede siamesa implementada, foram realizados ao todo $3 \times 3 = 9$ experimentos computacionais neste TCC. Usamos três bases de dados D_1 , D_2 e D_3 , onde D_2 e D_3 foram criadas via técnicas de pré-processamento mais drásticas e, para cada uma delas, variamos de três maneiras diferentes a variável “tamanho máximo de sequência” (*max_seq_length*).

Para o treinamento e a validação do nosso modelo usamos a partição 70/30, onde 70% do conjunto de dados foram alocados para treino e os outros 30% para validação. Um modelo comum de particionamento dos dados é ter um terceira partição menor do mesmo conjunto de dados para testes. Entretanto, neste trabalho foi adotada a mesma metodologia de partição dos dados da implementação da MaLSTM, que utilizou um conjunto de dados totalmente independente e novo, ou seja, que possuía sentenças de textos jamais vistas nas fases de treinamento e validação. A Figura 36 ilustra como os nossos dados de treinamento, validação e teste foram organizados.

Figura 36 – Partição das bases de dados utilizadas nas fases de treinamento, validação e testes.



Fonte: Elaborado pelo autor (2021).

O principal trecho de código dos experimentos computacionais realizados no trabalho está mostrado na Figura 37. Note que o trecho é uma função, que é responsável por

interfacear com o módulo *bootstrap* da aplicação (*main.py*) e controlar o fluxo principal dos experimentos. Note também que esta função recebe um único parâmetro/argumento chamado *n_rounds*, que corresponde ao número de iterações/rodadas que são executadas para cada experimento no intuito de ponderar os resultados obtidos nas fases de treinamento, validação e teste. Neste TCC definimos esse valor como 20, ou seja, foram realizadas 20 iterações por experimento e foi calculada a média aritmética dos resultados de cada experimento.

Figura 37 – Função principal de execução dos experimentos computacionais do módulo desenvolvido da aplicação chamado de *experiments.py*.

```
def run_experiments(n_rounds):
    configs_per_datasets = __get_defined_configs_per_dataset_type()

    for dataset_type in list(DatasetType):
        for config_dataset in configs_per_datasets[dataset_type]:
            n_epochs = config_dataset[0]
            max_seq_length = config_dataset[1]

            for round_number in range(1, n_rounds + 1, 1):
                hyperparameters = __get_defined_lstm_hyperparameters(n_epochs, max_seq_length, 1)
                model = __run_training_process_round(round_number, hyperparameters, dataset_type)
                __run_prediction_process_round(round_number, model, hyperparameters, dataset_type)

                __save_training_plot_performance_graph(n_rounds, dataset_type, max_seq_length, n_epochs)
                __save_prediction_authors_matrix_combinations(n_rounds, dataset_type, n_epochs, max_seq_length)
```

Fonte: Elaborado pelo autor (2021).

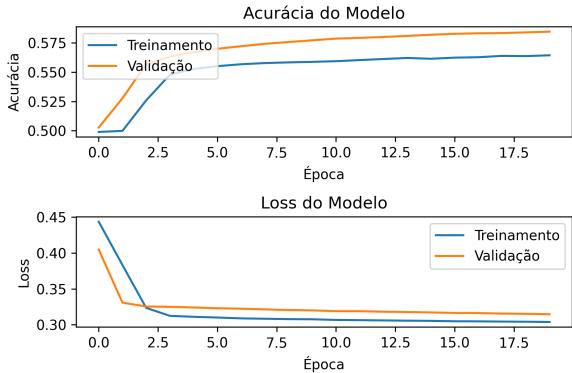
As seções seguintes apresentarão os resultados dos experimentos obtidos da média das 20 execuções (iterações) para cada tamanho máximo de sequência dos três *conjuntos de dados* (D_1 , D_2 e D_3) utilizados no trabalho. Para a fase de treinamento serão analisados os resultados finais a partir dos gráficos de curva de treinamento e validação. Já para a fase de teste serão analisados os resultados finais de predição a partir das tabelas de similaridades médias entre as combinações de sentenças dos autores selecionados.

4.1 RESULTADOS OBTIDOS COM O CONJUNTO DE DADOS D_1

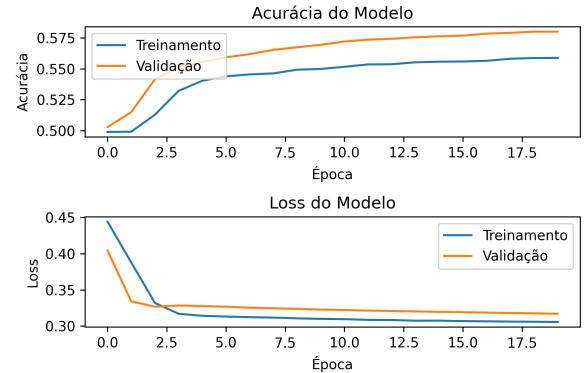
A Figura 38 corresponde aos gráficos de acurácia e *loss* das curvas de treinamento e validação resultante da média das 20 execuções realizadas para cada tamanho máximo de sequência selecionado do *conjunto de dados* D_1 , os quais são: 17 (média aritmética), 10 (mediana) e 14 (média aritmética entre média e mediana). De antemão é importante salientar que conforme mostrado na Tabela 9 o *conjunto de dados* D_1 possui os maiores tamanhos máximos da sequência definidos em comparação aos *conjuntos de dados* D_2 e D_3 devido conter sentenças que foram aplicadas somente pré-processamentos textuais básicos.

Figura 38 – Gráficos de acurácia e *loss* do treinamento e validação do *conjunto de dados* D_1 .

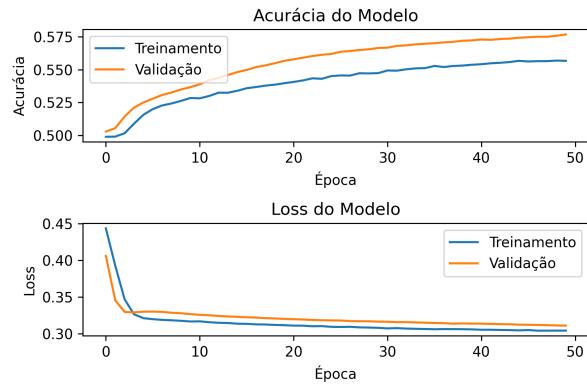
(a) 17 de tamanho da sequência com 20 épocas



(b) 14 de tamanho da sequência com 20 épocas



(c) 10 de tamanho da sequência com 50 épocas



Fonte: Elaborado pelo autor (2021).

Note que os resultados mostrados na Figura 38 são visualmente idênticos, porém analisando os dados a fundo há uma leve vantagem de poucos décimos na acurácia e *loss* do treinamento com tamanho da sequência 17 em relação aos outros tamanhos. Outro ponto a ser considerado é que o treinamento com tamanho da sequência 10 foi realizado com maior número de épocas (50) em comparação aos outros, pois somente com 20 épocas notamos que a rede ainda tinha capacidade de aprender, dado que a curva de acurácia ainda apresentava tendência de crescimento.

Os testes pós treinamento da rede para cada tamanho máximo de sequência foram realizados utilizando o conjunto de dados D_1 de teste e geraram as seguintes tabelas de similaridades que contém as médias dos índices de similaridade de Manhattan entre os pares de sentenças dos autores selecionados. Quando autores de um par são o mesmo esperava-se que similaridade apresentasse valores maiores do que aqueles que correspondem a autores distintos, o que se confirmou.

Tabela 12 – Tabela de similaridades médias entre combinações de sentenças de autores da base de dados D_1 usando tamanho da sequência 17.

	Faulkner	Hemingway	Roth
Faulkner	0.8329 ± 0.0139	0.6780 ± 0.0103	0.5396 ± 0.0123
Hemingway	0.6780 ± 0.0103	0.7460 ± 0.0100	0.6570 ± 0.0159
Roth	0.5396 ± 0.0123	0.6570 ± 0.0159	0.7332 ± 0.0104

Fonte: Elaborado pelo autor (2021).

Tabela 13 – Tabela de similaridades médias entre combinações de frases de autores da base de dados D_1 usando tamanho da sequência 14.

	Faulkner	Hemingway	Roth
Faulkner	0.8092 ± 0.0150	0.6648 ± 0.0148	0.5489 ± 0.0316
Hemingway	0.6648 ± 0.0148	0.7347 ± 0.0091	0.6682 ± 0.0198
Roth	0.5489 ± 0.0316	0.6682 ± 0.0198	0.7434 ± 0.0110

Fonte: Elaborado pelo autor (2021).

Tabela 14 – Tabela de similaridades médias entre combinações de frases de autores da base de dados D_1 usando tamanho da sequência 10.

	Faulkner	Hemingway	Roth
Faulkner	0.8083 ± 0.0289	0.6750 ± 0.0151	0.5525 ± 0.0295
Hemingway	0.6750 ± 0.0151	0.7509 ± 0.0095	0.7022 ± 0.0100
Roth	0.5525 ± 0.0295	0.7022 ± 0.0100	0.7431 ± 0.0087

Fonte: Elaborado pelo autor (2021).

Analisando os resultados das Tabelas 12, 13 e 14, verificamos que a rede neural apresentou melhor performance na identificação dos próprios autores, como destacado em negrito em suas diagonais principais. Todavia, a rede também apresentou uma grande taxa de erros ao fornecer similaridades relativamente altas entre pares de frases de autores diferentes.

O modelo original que serviu de inspiração a este trabalho de conclusão apresenta melhor performance voltada para determinação de similaridade entre pares de frases em que características sintáticas possuem maior relevância do que características semânticas para extração de significados.

Neste trabalho, como buscamos a extração de traços **semânticos** literários entre autores, o desempenho não foi o mesmo do que o obtido no desafio *Kaggle*, descrito no início deste capítulo.

Podemos ainda inferir que os melhores resultados estão presentes nos testes com tamanhos da sequência 17 e 14, respectivamente. Porém nota-se uma diferença mínima na perfor-

mance do modelo ao usarmos os três tamanhos da sequência adotados para este conjunto de dados.

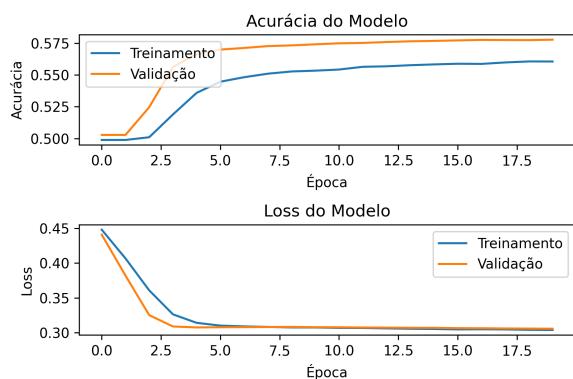
Este fato corrobora para com os resultados obtidos no processo de treinamento da rede, em que o tamanho da sequência 17 forneceu à rede resultados ligeiramente superiores aos demais treinados. Outro ponto a ser destacado são os baixos valores de desvio padrão observados em todos os cenários apresentados nas tabelas de similaridade, ilustrando que as 20 execuções dos testes para cada tamanho da sequência geraram resultados bem próximos, dada a baixa dispersão dos resultados obtidos.

4.2 RESULTADOS OBTIDOS COM O CONJUNTO DE DADOS D_2

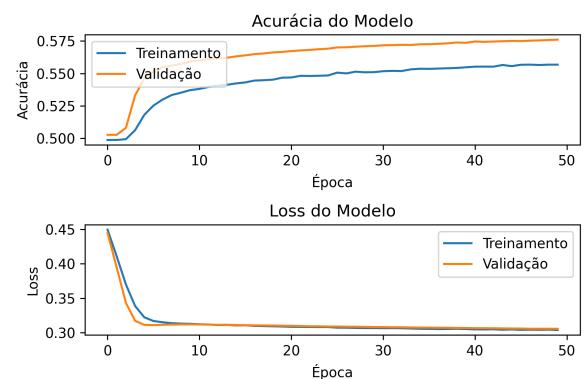
O conjunto de dados D_2 , que contém as sentenças de texto sem *stopwords*, foi utilizado em nossos experimentos com os três tamanhos de sequência: 9 (média aritmética), 5 (mediana) e 7 (média aritmética entre média e mediana). Na Figura 39 estão apresentados os gráficos de acurácia e *loss* das curvas de treinamento e validação gerados a partir da média das 20 execuções para cada tamanho da sequência definido.

Figura 39 – Gráficos de acurácia e *loss* do treinamento e validação do conjunto de dados D_2 .

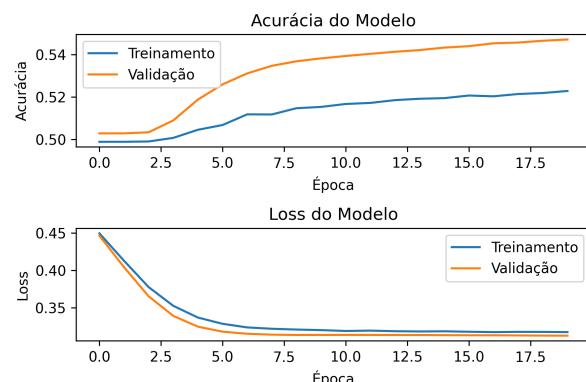
(a) 9 de tamanho da sequência com 20 épocas



(b) 7 de tamanho da sequência com 50 épocas



(c) 5 de tamanho da sequência com 20 épocas



Fonte: Elaborado pelo autor (2021).

A partir dos gráficos mostrados acima notamos que, similarmente ao caso do conjunto de dados D_1 , os melhores resultados de acurácia e *loss* são referentes ao treinamento com o maior tamanho máximo de sequência, que no caso é 9, mas sem diferenças consideráveis em comparação aos outros tamanhos.

Outro ponto a ser observado é que o tamanho de sequência igual a 7 ainda tinha uma gradativa capacidade de aprender após 20 épocas. Utilizamos, então, 50 épocas, e o modelo foi capaz de alcançar praticamente os mesmos resultados de acurácia do tamanho da sequência 9. Usando mais de 50 épocas para treinamento da rede, observou-se um início de sobretreinamento.

Os resultados dos testes para cada tamanho de sequência realizados usando o conjunto de dados D_2 de teste estão elencados nas Tabelas 15, 16 e 17.

Tabela 15 – Tabela de similaridades médias entre combinações de frases de autores do conjunto de dados D_2 usando tamanho da sequência 9.

	Faulkner	Hemingway	Roth
Faulkner	0.8103 ± 0.0124	0.7049 ± 0.0111	0.6024 ± 0.0188
Hemingway	0.7049 ± 0.0111	0.7378 ± 0.0066	0.6758 ± 0.0064
Roth	0.6024 ± 0.0188	0.6758 ± 0.0064	0.7196 ± 0.0144

Fonte: Elaborado pelo autor (2021).

Tabela 16 – Tabela de similaridades médias entre combinações de frases de autores do conjunto de dados D_2 usando tamanho da sequência 7.

	Faulkner	Hemingway	Roth
Faulkner	0.7974 ± 0.0199	0.7098 ± 0.0095	0.6113 ± 0.0162
Hemingway	0.7098 ± 0.0095	0.7253 ± 0.0112	0.6804 ± 0.0089
Roth	0.6113 ± 0.0162	0.6804 ± 0.0089	0.7417 ± 0.0251

Fonte: Elaborado pelo autor (2021).

Tabela 17 – Tabela de similaridades médias entre combinações de frases de autores do conjunto de dados D_2 usando tamanho da sequência 5.

	Faulkner	Hemingway	Roth
Faulkner	0.7279 ± 0.0352	0.7133 ± 0.0237	0.6636 ± 0.0339
Hemingway	0.7133 ± 0.0237	0.7086 ± 0.0245	0.6697 ± 0.0257
Roth	0.6636 ± 0.0339	0.6697 ± 0.0257	0.7243 ± 0.0258

Fonte: Elaborado pelo autor (2021).

Ao analisarmos os resultados dos testes a partir das tabelas acima, notamos que semelhantemente aos resultados apresentados no caso do conjunto de dados D_1 , o modelo apresentou melhor performance na identificação de mesmos autores, como mostrado nas diagonais

principais, com exceção do teste com tamanho da sequência 5 ilustrado na Tabela 17, onde a combinação das frases do autor *Hemingway* com ele próprio teve um valor médio de similaridade levemente menor do que a média de similaridade dos autores *Hemingway* com *Faulkner*. Acreditamos que o motivo disto ter ocorrido é tanto pela remoção de *stopwords* do conjunto sentenças dos autores que compõem o conjunto de dados D_2 como ao baixo valor definido ao tamanho da sequência, pois ambos impactam negativamente na capacidade da rede de aprender os traços semânticos literários dos autores.

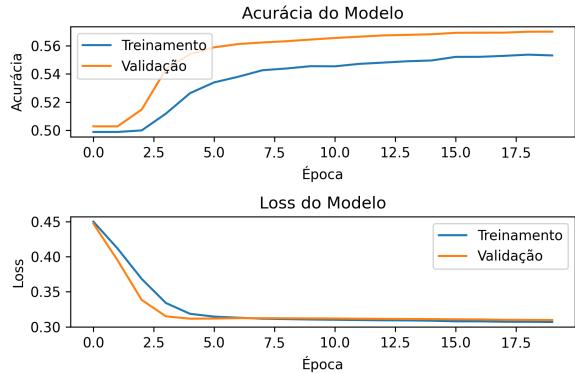
Também é perceptível que o tamanho da sequência 9 possui ligeiramente os melhores resultados de predição, o que acaba condizendo com os resultados obtidos na fase de treinamento do conjunto de dados D_2 . Além disso, novamente comparando o desempenho do modelo com resultados do conjunto de dados D_1 , os desvios padrões das previsões foram baixos, evidenciando que os resultados das 20 execuções realizadas na fase de treinamento tiveram pequenas diferenças, o que demonstra consistência na rede.

4.3 RESULTADOS OBTIDOS COM O CONJUNTO DE DADOS D_3

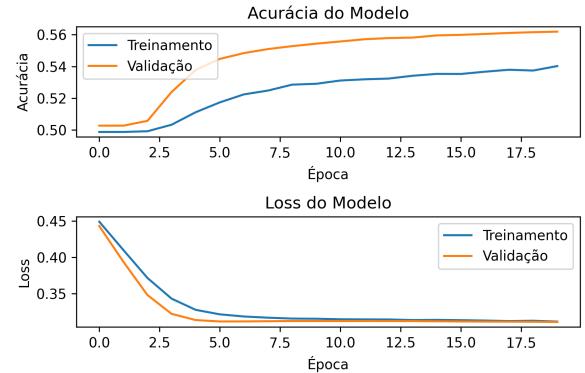
O conjunto de dados D_3 é bem parecido com o conjunto de dados D_2 , com a principal exceção que além de ser aplicado a remoção de *stopwords* também é aplicada a técnica de lematização das palavras das sentenças de textos presentes em seu conjunto de dados. Assim, como a lematização não remove nenhuma das palavras de uma sentença de texto, mas sim as reduzem para a sua forma canônica/lematizada, os tamanhos máximos da sequência definidos para este conjunto de dados é exatamente o mesmo do que para o conjunto de dados D_2 , os quais são: 9 (média aritmética), 5 (mediana) e 7 (média aritmética entre média e mediana). A Figura 40 mostra os resultados obtidos da fase de treinamento da rede por meio dos gráficos de acurácia e *loss* das curvas de treinamento e validação gerados pela média das 20 execuções para cada tamanho da sequência estabelecido para o conjunto de dados D_3 .

Figura 40 – Gráficos de acurácia e *loss* do treinamento e validação no conjunto de dados D_3 .

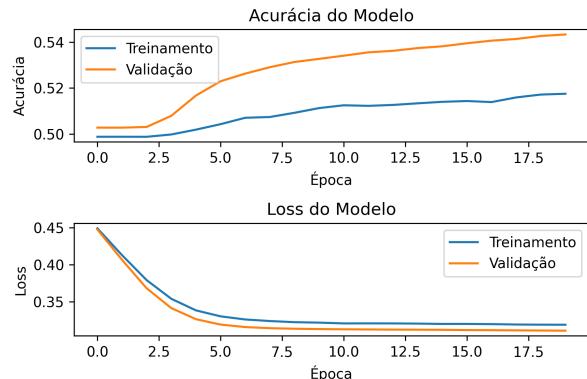
(a) 9 de tamanho da sequência com 20 épocas



(b) 7 de tamanho da sequência com 20 épocas



(c) 5 de tamanho da sequência com 20 épocas



Fonte: Elaborado pelo autor (2021).

Analisando os gráficos da Figura 40 percebemos que os resultados de treinamento no conjunto de dados D_3 são pouco inferiores, mas bem similares aos obtidos nos conjuntos de dados D_1 e D_2 , especialmente ao conjunto de dados D_2 por conter os mesmos tamanhos de sequência e sentenças de texto. Os gráficos também mostram que para todos os tamanhos da sequência treinados foram utilizados 20 épocas. Com mais épocas observava-se o início de sobretreinamento da rede.

Abaixo estão as tabelas que contém os resultados das similaridades médias dos testes de cada tamanho da sequência no conjunto de dados D_3 de teste.

Tabela 18 – Tabela de similaridades médias entre combinações de frases de autores no conjunto de dados D_3 usando tamanho da sequência 9.

	Faulkner	Hemingway	Roth
Faulkner	0.7909 \pm 0.0241	0.7055 \pm 0.0159	0.6152 \pm 0.0283
Hemingway	0.7055 \pm 0.0159	0.7321 \pm 0.0102	0.6910 \pm 0.0140
Roth	0.6152 \pm 0.0283	0.6910 \pm 0.0140	0.7313 \pm 0.0142

Fonte: Elaborado pelo autor (2021).

Tabela 19 – Tabela de similaridades médias entre combinações de frases de autores no conjunto de dados D_3 usando tamanho da sequência 7.

	Faulkner	Hemingway	Roth
Faulkner	0.7728 ± 0.0292	0.7169 ± 0.0202	0.6464 ± 0.0306
Hemingway	0.7169 ± 0.0202	0.7150 ± 0.0183	0.6939 ± 0.0222
Roth	0.6464 ± 0.0306	0.6939 ± 0.0222	0.7480 ± 0.0236

Fonte: Elaborado pelo autor (2021).

Tabela 20 – Tabela de similaridades médias entre combinações de frases de autores no conjunto de dados D_3 usando tamanho da sequência 5.

	Faulkner	Hemingway	Roth
Faulkner	0.7306 ± 0.0176	0.7274 ± 0.0265	0.6653 ± 0.0301
Hemingway	0.7274 ± 0.0265	0.7069 ± 0.0216	0.6613 ± 0.0352
Roth	0.6653 ± 0.0301	0.6613 ± 0.0352	0.7252 ± 0.0221

Fonte: Elaborado pelo autor (2021).

Com os resultados apresentados nas tabelas dos testes realizados logo acima, percebemos que novamente as diagonais principais, que são referentes as combinações entre os mesmos autores, apresentaram os melhores resultados. Todavia, como ocorrido na Tabela 17, referente ao conjunto de dados D_2 , há duas exceções nos melhores resultados de similaridade média das diagonais principais que são nos testes com tamanho da sequência 7 e 5. Em ambos observou-se a mesma situação que no conjunto de dados D_2 , onde as frases do autor *Hemingway* com ele próprio também tiveram valores de similaridades média levemente menor do que as similaridades médias dos autores *Hemingway* com *Faulkner*. Além dos motivos já apontados na análise dos resultados no conjunto de dados D_2 , também acreditamos que pelo fato dos autores *Hemingway* e *Faulkner* serem de um mesmo movimento literário artístico, possuírem a mesma nacionalidade e viverem literalmente em um mesmo período histórico a rede não foi capaz de distinguir bem as *features* dos traços semânticos literários dos dois autores, além de outros motivos que não temos conhecimento de causa para nos embasarmos.

Podemos deduzir também que, assim como no conjunto de dados D_2 , o tamanho da sequência 9 contém os melhores resultados de similaridades médias entre as combinações de frases dos autores.

4.4 ANÁLISES FINAIS DOS RESULTADOS

Como destacado no capítulo anterior, o conjunto de dados D_1 é caracterizado por conter apenas pré-processamento textual básico, tais como: remoção de pontuações e caracteres numéricos, normalização das palavras para letras minúsculas e afins. Assim, nenhuma

palavra das sentenças de textos extraídas das obras dos autores é de fato perdida, o que difere do caso dos conjunto de dados D_2 e D_3 , onde são aplicadas técnicas de pré-processamento que removem ou alteram as palavras das sentenças podendo retirar características essenciais de escritas que diferem os autores.

Consequentemente, acreditamos que isto limita a capacidade da rede de performar melhor na tarefa de extrair as *features* dos traços semânticos literários dos autores. E isto fica evidente com os melhores resultados dos testes apresentados nas tabelas do conjunto de dados D_1 em comparação aos resultados nos conjunto de dados D_2 e D_3 .

Uma outra hipótese que também acreditamos que impactou negativamente na performance do modelo, foi que todos os autores escolhidos para compor os conjunto de dados do trabalho são de um mesmo estilo literário e praticamente de um mesmo período temporal, especialmente *Faulkner* e *Hemingway* que nasceram e morreram aproximadamente na mesma época. Isto ficou evidenciado nos resultados dos testes apresentados nas tabelas de similaridades médias das seções anteriores, em que os valores de similaridades que correspondem nas combinações das frases entre autores diferentes tiveram valores relativamente altos.

O impacto no desempenho do modelo é ainda maior quando os tamanhos da sequência são levados em consideração, pois em todos os experimentos realizados os melhores resultados para cada conjunto de dados foram aqueles que tiveram o maior tamanho máximo de sequência.

O desempenho do modelo na comparação de pares por sentenças também pode ter sido prejudicado pelo contexto da língua inglesa em si, gramaticalmente mais simples do que outras, de origem latina, por exemplo.

Podemos inferir que a rede siamesa implementada tendo como estratégia a comparação de pares de sentenças não foi capaz de extrair as melhores *features* que determinassem de fato os traços essenciais de cada autor. É importante notar que o modelo performa bem quando a exigência de interpretação de nível semântico é baixa, todavia a captura do *ethos* literário de cada autor é algo muito mais subjetivo, difícil e escapadiço para as máquinas.

No intuito de melhorar a performance da rede de forma que ela extraia melhores *features* uma mudança de estratégia seria algo necessário.

- Uma das possibilidades seria a aplicação de uma metodologia ou técnica de extração de características antes de aplicar o *word embedding*.
- Outra possibilidade que também acreditamos que poderia melhorar a performance do modelo seria ao invés de compararmos palavra por palavra usando o *Word2Vec*,

comparássemos parágrafo por parágrafo usando Paragraph2Vec ou sentença por sentença usando o Sentence2Vec. A hipótese é que trechos mais longos de texto podem trazer em si mais informações na busca por capturar os relacionamentos semânticos.

- Realizar tipos de pré-processamento mais drásticos parece não ser uma boa opção, pois retira informações que seriam importantes para que o modelo aprenda. A nossa ideia é manter uma abordagem com dados brutos, sem remover nada.
- Acreditamos que usar uma estratégia de *word embedding* contextual pode ajudar a melhorar a performance da rede nesta tarefa. Um exemplo seria utilizar o *Sentence-BERT*, pois, afinal de contas a tarefa exige mais do que representar palavras, mas representar palavras em diferentes contextos.
- Uma conjectura que parece plausível é testar valores maiores para a variável “tamanho de sequência”.
- Pensamos também em utilizar uma CNN para extração de features, a partir delas realizar um *word embedding* e alimentar uma RNS Bi-LSTM, para se levar em consideração vizinhanças de palavras tanto para a esquerda como para a direita.
- Algo que também nos parece plausível é criar uma base de dados personalizada, por exemplo composta apenas por substantivos predominantes, adjetivos ou alguma outra característica que um especialista de causa poderia nos indicar a partir de sua expertise.

5 CONSIDERAÇÕES FINAIS

Neste TCC foram analisados os resultados da performance de uma rede neural siamesa LSTM na tarefa de obtenção da medida de similaridade de Manhattan entre pares de sentenças de autores renomados da literatura norte-americana: *Faulkner*, *Hemingway* e *Roth*.

A extração de características dos autores foram realizadas por meio da sentenças de texto provenientes de suas obras literárias que foram estruturadas em pares de frases formando três conjuntos de dados distintos variando em relação ao tipo de pré-processamento aplicado, os quais são: D_1 , contendo sentenças de texto com pré-processamento básico, D_2 , com aplicação de pré-processamento básico e remoção de *stopwords* e por fim o D_3 , que é similar ao D_2 , mas também com a aplicação da técnica de lematização de palavras.

Para realização do trabalho foi necessário o estudo de diversos conceitos da área de inteligência artificial e de aprendizado de máquina. Dentre os principais estão: processamento da linguagem natural (PLN), similaridade de textos, medidas de similaridade, em particular a medida de similaridade de *Manhattan*, redes neurais profundas, especialmente as redes recorrentes LSTM e siamesas, técnicas de *word embeddings* e seus algoritmos, técnicas de pré-processamento e de otimização de hiperparâmetros, entre diversas técnicas de programação.

Todos esses conceitos, metodologias, recursos e técnicas foram essenciais para o sucesso da criação e estruturação dos conjuntos de dados utilizados, a construção do modelo de aprendizagem e a realização dos experimentos computacionais.

Os experimentos computacionais mostraram que a rede siamesa implementada com base na MaLSTM, que foi utilizada para resolver o problema do desafio do *Kaggle*, não performou bem na tarefa de determinar a similaridade entre pares de sentenças literárias.

Posto isto, pode ser que a estratégia de comparação par a par seja melhor em cenários em que o nível necessário de interpretação sintática é maior que a semântica, que não é o caso dos estudos literários realizados neste TCC.

Em suma, este TCC contribuiu para o estudo investigativo acerca das redes siamesas na tarefa de determinação de similaridades entre pares de sentenças literárias. Todos os objetivos estabelecidos neste TCC foram alcançados, conforme o planejado. Acreditamos ainda que fomos além, ao lançar luz sobre uma série de possibilidades que se colocam a partir dos resultados obtidos e analisados.

5.1 TRABALHOS FUTUROS

A utilização de novas estratégias, metodologias, assim como novas tecnologias com intuito de melhorar o desempenho do modelo de aprendizado e resolver o problema proposto neste trabalho de conclusão podem ser investigados e aplicados em trabalhos futuros. Dentre eles estão:

- Resolver o problema proposto usando uma abordagem tradicional de classificação multiclasses através de arquiteturas de rede neurais profundas conhecidas, tais como: CNNs, LSTMs, GRUs ou outras;
- Utilizar um extrator de *features* ou um redutor de dimensionalidade antes de aplicar a técnica de *word embedding* nas palavras ou sentenças de texto retiradas das obras dos autores;
- Aplicar outras estratégias de *word embeddings*, tais como: Sentence2Vec, Doc2Vec, Paragraph2Vec e SentenceBERT;
- Aplicar outras ferramentas de otimização de hiperparâmetros que usam algoritmo Bayesiano com intuito de melhorar os ajustes de hiperparâmetricos da rede neural utilizada;
- Modificar a arquitetura das LSTMs acopladas na RNS.
- Personalizar uma base de dados levando-se em conta a expertise de algum especialista de causa;
- Utilizar uma rede convolucional para extrair *features* e uma rede Bi-LSTM para determinar as medidas de similaridade;
- Testar diferentes modelos munidos de outras medidas de similaridade.

REFERÊNCIAS

- ABIODUN, O. I. et al. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, v. 4, n. 11, p. e00938, 2018. ISSN 2405-8440. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S2405844018332067>>.
- ACADEMY, D. S. *Deep Learning Book*. 2021. Disponível em: <<https://www.deeplearningbook.com.br/>>. Acesso em: 01 mai. 2021.
- ALMEIDA, F.; XEXÉO, G. Word embeddings: A survey. *arXiv preprint arXiv:1901.09069*, 2019.
- ALOM, M. Z. et al. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, v. 8, p. 292, 03 2019.
- ALRUBAN, A. *Hyper parameters and their importance in deep learning?* 2019.
- AMARI, S.-i.; ARBIB, M. A. *Competition and Cooperation in Neural Nets: Proceedings of the US-Japan Joint Seminar Held at Kyoto, Japan February 15–19, 1982*. [S.l.]: Springer Science & Business Media, 2013. v. 45.
- ARASANIPALAI, A. U. *How To Make Deep Learning Models That Don't Suck*. 2021. Disponível em: <<https://nanonets.com/blog/hyperparameter-optimization/>>. Acesso em: 09 set. 2021.
- BAEZA-YATES, R.; RIBEIRO-NETO, B. et al. *Modern information retrieval*. [S.l.]: ACM press New York, 1999. v. 463.
- BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- BALAKRISHNAN, V.; LLOYD-YEMOH, E. Stemming and lemmatization: A comparison of retrieval performances. 2014.
- BARRÓN-CEDEÑO, A.; GUPTA, P.; ROSSO, P. Methods for cross-language plagiarism detection. *Knowledge-Based Systems*, Elsevier, v. 50, p. 211–217, 2013.
- BARZILAY, R.; ELHADAD, M. Using lexical chains for text summarization. *Advances in automatic text summarization*, p. 111–121, 1999.
- BASHAR, A. Survey on evolving deep learning neural network architectures. *Journal of Artificial Intelligence and Capsule Networks*, v. 2019, p. 73–82, 12 2019.
- BEKMIRZAEV, S.; KIM, T.-H.; LEE, B.-C. Pairwise similarity analysis and quality estimation on classical chinese poetry of ancient korea in 15th century. *International Journal of Applied Engineering Research*, v. 12, n. 23, p. 13884–13890, 2017.
- BENGIO, Y. et al. A neural probabilistic language model. *The journal of machine learning research*, JMLR. org, v. 3, p. 1137–1155, 2003.
- BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, IEEE, v. 5, n. 2, p. 157–166, 1994.

- BERGMANIS, T.; GOLDWATER, S. Context sensitive neural lemmatization with lematus. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. [S.l.: s.n.], 2018. p. 1391–1400.
- BERLEMONT, S. et al. Class-balanced siamese neural networks. *Neurocomputing*, Elsevier, v. 273, p. 47–56, 2018.
- BISSUEL, A. *Hyper-parameter optimization algorithms: a short review*. 2019. Disponível em: <<https://medium.com/criteo-engineering/hyper-parameter-optimization-algorithms-2fe447525903>>. Acesso em: 09 set. 2021.
- BORGES, V. *Aplicações de Processamento de Linguagem Natural importantes para sua empresa*. 2018. Disponível em: <<https://www.viacognitiva.com.br/single-post/aplicacoes-PLN-importante>>. Acesso em: 07 jun. 2020.
- BROMLEY, J. et al. Signature verification using a "siamese" time delay neural network. *Advances in neural information processing systems*, v. 6, p. 737–744, 1993.
- BROWN, P. F. et al. Word-sense disambiguation using statistical methods. In: *29th Annual meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 1991. p. 264–270.
- BROWNLEE, J. *What Are Word Embeddings for Text?* 2019. Disponível em: <<https://machinelearningmastery.com/what-are-word-embeddings/>>. Acesso em: 19 jun. 2021.
- BROWNLEE, J. *What is the Difference Between a Parameter and a Hyperparameter?* 2019. Disponível em: <<https://machinelearningmastery.com/difference-between-a-parameter-and-a-hyperparameter/>>. Acesso em: 13 aug. 2021.
- BROWNLEE, J. *Why Training a Neural Network Is Hard*. 2019. Disponível em: <<https://machinelearningmastery.com/why-training-a-neural-network-is-hard/#:~:text=The%20iterative%20training%20process%20of,examples%20in%20the%20training%20dataset.>> Acesso em: 01 mai. 2021.
- BROWNLEE, J. *4 Distance Measures for Machine Learning*. 2020. Disponível em: <<https://machinelearningmastery.com/distance-measures-for-machine-learning/>>. Acesso em: 26 ago. 2021.
- BROWNLEE, J. *Hyperparameter Optimization With Random Search and Grid Search*. 2020. Disponível em: <<https://machinelearningmastery.com/hyperparameter-optimization-with-random-search-and-grid-search/>>. Acesso em: 09 set. 2021.
- BÖHM, T. *A first Introduction to SELUs and why you should start using them as your Activation Functions*. 2018. Disponível em: <<https://towardsdatascience.com/gentle-introduction-to-selus-b19943068cd9>>. Acesso em: 08 set. 2021.
- CARVALHO, A. P. de Leon F. de. *Redes Neurais Artificiais*. 2009. Disponível em: <<https://sites.icmc.usp.br/andre/research/neural/#intro>>. Acesso em: 29 abr. 2021.

- CHARNIAK, E. Passing markers: A theory of contextual influence in language comprehension. *Cognitive science*, Elsevier, v. 7, n. 3, p. 171–190, 1983.
- CHAUHAN, N. S. *Hyperparameter Optimization for Machine Learning Models*. 2020. Disponível em: <<https://www.kdnuggets.com/2020/05/hyperparameter-optimization-machine-learning-models.html>>. Acesso em: 09 set. 2021.
- CHEN, J. *Neural Network*. 2020. Disponível em: <<https://www.investopedia.com/terms/n/neuralnetwork.asp>>. Acesso em: 29 abr. 2021.
- CHI, Z.; ZHANG, B. A sentence similarity estimation method based on improved siamese network. *Journal of Intelligent Learning Systems and Applications*, Scientific Research Publishing, v. 10, n. 4, p. 121–134, 2018.
- CHICCO, D. Siamese neural networks: An overview. *Artificial Neural Networks*, Springer, p. 73–94, 2021.
- CHITRAO, M. V.; GRISHMAN, R. Statistical parsing of messages. In: *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*. [S.l.: s.n.], 1990.
- CHOMSKY, N. Logical structures in language. *American Documentation (pre-1986)*, Wiley Periodicals Inc., v. 8, n. 4, p. 284, 1957.
- CHOPRA, S.; HADSELL, R.; LECUN, Y. Learning a similarity metric discriminatively, with application to face verification. In: IEEE. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. [S.l.], 2005. v. 1, p. 539–546.
- COELHO, C. *Um guia completo para o pré-processamento de dados em machine learning*. 2019. Disponível em: <[https://caiquecoelho.medium.com/um-guia-completo-para-o-pr% C3%A9-processamento-de-dados-em-machine-learning-f860fbadabe1](https://caiquecoelho.medium.com/um-guia-completo-para-o-pr%C3%A9-processamento-de-dados-em-machine-learning-f860fbadabe1)>. Acesso em: 29 mai. 2021.
- COHEN, E. *How to predict Quora Question Pairs using Siamese Manhattan LSTM*. 2017. Disponível em: <<https://blog.mlreview.com/implementing-malstm-on-kaggles-quora-question-pairs-competition-8b31b0b16a07>>. Acesso em: 20 jun. 2021.
- COLLINS, A. M.; QUILLIAN, M. R. Experiments on semantic memory and language comprehension. John Wiley & Sons, 1972.
- COLLOBERT, R.; WESTON, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In: *Proceedings of the 25th international conference on Machine learning*. [S.l.: s.n.], 2008. p. 160–167.
- CRAPAROTTA, G.; THOMASSEY, S.; BIOLATTI, A. A siamese neural network application for sales forecasting of new fashion products using heterogeneous data. *International Journal of Computational Intelligence Systems*, Atlantis Press, v. 12, n. 2, p. 1537–1546, 2019.
- CROFT, W. B.; METZLER, D.; STROHMAN, T. *Search engines: Information retrieval in practice*. [S.l.]: Addison-Wesley Reading, 2010. v. 520.

- CUI, W. et al. Face recognition via convolutional neural networks and siamese neural networks. In: IEEE. *2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*. [S.I.], 2019. p. 746–750.
- CUI, Y.; WANG, S.; LI, J. Lstm neural reordering feature for statistical machine translation. *arXiv preprint arXiv:1512.00177*, 2015.
- DAUD, A. et al. Finding rising stars through hot topics detection. *Future Generation Computer Systems*, v. 115, p. 798–813, 2021. ISSN 0167-739X. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0167739X20329903>>.
- DETTMERS, T. *Deep Learning in a Nutshell: History and Training*. 2015. Disponível em: <<https://developer.nvidia.com/blog/deep-learning-nutshell-history-training/>>. Acesso em: 05 mai. 2021.
- DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- EDUCATION, I. C. *Neural Networks*. 2020. Disponível em: <<https://www.ibm.com/cloud/learn/neural-networks>>. Acesso em: 01 mai. 2021.
- EL-KHAIR, I. A. Effects of stop words elimination for arabic information retrieval: A comparative study. *International Journal of Computing Information Sciences*, v. 4, p. 119–133, 01 2006.
- ELMAN, J. L. Finding structure in time. *Cognitive science*, Wiley Online Library, v. 14, n. 2, p. 179–211, 1990.
- FILLMORE, C. J.; FILLMORE, S. Case grammar. *Universals in Linguistic Theory*. Holt, Rinehart, and Winston: New York, p. 1–88, 1968.
- FRAZIER, P. I. *A Tutorial on Bayesian Optimization*. 2018.
- GAVRILOVA, Y. *A Guide to Deep Learning and Neural Networks*. 2020. Disponível em: <<https://serokell.io/blog/deep-learning-and-neural-network-guide>>. Acesso em: 03 mai. 2021.
- GHOSH, S. et al. A novel spatio-temporal siamese network for 3d signature recognition. *Pattern Recognition Letters*, Elsevier, v. 144, p. 13–20, 2021.
- GOHRANI, K. *Metrics used in Machine Learning*. 2019. Disponível em: <https://medium.com/@kunal_gohrani/different-types-of-distance-metrics-used-in-machine-learning-e9928c5e26c7>. Acesso em: 26 ago. 2021.
- GOMAA, W. H.; FAHMY, A. A. et al. A survey of text similarity approaches. *International Journal of Computer Applications*, Citeseer, v. 68, n. 13, p. 13–18, 2013.
- GOMES, P. C. T. *Conheça as Etapas do Pré-Processamento de dados*. 2019. Disponível em: <<https://www.datageeks.com.br/pre-processamento-de-dados/>>. Acesso em: 29 mai. 2021.
- GONZALEZ, M.; LIMA, V. L. S. Recuperação de informação e processamento da linguagem natural. In: *XXIII Congresso da Sociedade Brasileira de Computação*. [S.I.: s.n.], 2003. v. 3, p. 347–395.

- GOZZOLI, A. *Practical Guide to Hyperparameters Optimization for Deep Learning Models*. 2018. Disponível em: <<https://blog.floydhub.com/guide-to-hyperparameters-search-for-deep-learning-models/>>. Acesso em: 14 jun. 2021.
- GRAVES, A.; FERNÁNDEZ, S.; SCHMIDHUBER, J. Bidirectional lstm networks for improved phoneme classification and recognition. In: SPRINGER. *International conference on artificial neural networks*. [S.l.], 2005. p. 799–804.
- GRAVES, A.; JAITLY, N.; MOHAMED, A.-r. Hybrid speech recognition with deep bidirectional lstm. In: IEEE. *2013 IEEE workshop on automatic speech recognition and understanding*. [S.l.], 2013. p. 273–278.
- GROSZ, B. J. et al. Team: An experiment in the design of transportable natural-language interfaces. *Artificial Intelligence*, Elsevier, v. 32, n. 2, p. 173–243, 1987.
- HAN, S. et al. Ese: Efficient speech recognition engine with sparse lstm on fpga. In: *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. [S.l.: s.n.], 2017. p. 75–84.
- HANSEN, C. *Activation Functions Explained - GELU, SELU, ELU, ReLU and more*. 2019. Disponível em: <<https://mlfromscratch.com/activation-functions-explained/>>. Acesso em: 08 set. 2021.
- HAYKIN, S. *Neural networks and learning machines*, 3/E. [S.l.]: Pearson Education India, 2010.
- HEBB, D. O. *The organization of behavior: A neuropsychological theory*. [S.l.]: New York: Wiley, 1949.
- HOCHREITER, S. Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, v. 91, n. 1, 1991.
- HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.
- HOPFIELD, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, v. 79, n. 8, p. 2554–2558, Apr 1982. ISSN 0027-8424. 6953413[pmid]. Disponível em: <<https://pubmed.ncbi.nlm.nih.gov/6953413/>>.
- HOUVARDAS, J.; STAMATATOS, E. N-gram feature selection for authorship identification. In: SPRINGER. *International conference on artificial intelligence: Methodology, systems, and applications*. [S.l.], 2006. p. 77–86.
- HOWARD, J.; RUDER, S. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.
- HUANG, Z.; XU, W.; YU, K. *Bidirectional LSTM-CRF Models for Sequence Tagging*. 2015.
- HUTCHINS, W. *Early Years in Machine Translation: Memoirs and Biographies of Pioneers*. J. Benjamins, 2000. (3]). ISBN 9789027245861. Disponível em: <<https://books.google.com.br/books?id=-GW8lOYl3AAC>>.

- HUTCHINS, W. J.; DOSTERT, L.; GARVIN, P. The georgetown-ibm experiment. In: CITESEER. *In.* [S.I.], 1955.
- ICHIDA, A. Y.; MENEGUZZI, F.; RUIZ, D. D. Measuring semantic similarity between sentences using a siamese neural network. In: IEEE. *2018 International Joint Conference on Neural Networks (IJCNN).* [S.I.], 2018. p. 1–7.
- IEEE. Ieee first annual international conference on neural networks san diego, california june 21-24, 1987. *IEEE Expert*, v. 2, n. 2, p. 14–14, 1987.
- J, S. B. *A friendly introduction to Siamese Networks*. 2020. Disponível em: <<https://towardsdatascience.com/a-friendly-introduction-to-siamese-networks-85ab17522942>>. Acesso em: 13 mai. 2021.
- JAGTAP, A. B. et al. Verification of genuine and forged offline signatures using siamese neural network (snn). *Multimedia Tools and Applications*, Springer, v. 79, n. 47, p. 35109–35123, 2020.
- JASSOVA, B. *Conversational AI Statistics: NLP Chatbots in 2020*. 2020. Disponível em: <<https://landbot.io/blog/conversational-ai-statistics/>>. Acesso em: 06 jun. 2020.
- JOULIN, A. et al. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.
- JR, J. G. C. Politics: Automated ideological reasoning. *Cognitive Science*, Elsevier, v. 2, n. 1, p. 27–51, 1978.
- KANG, E. *Long Short-Term Memory (LSTM): Concept*. 2017. Disponível em: <<https://medium.com/@kangeugine/long-short-term-memory-lstm-concept-cb3283934359>>. Acesso em: 11 mai. 2021.
- KARANI, D. *Introduction to Word Embedding and Word2Vec*. 2018. Disponível em: <<https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa>>. Acesso em: 19 jun. 2021.
- KE, W. et al. Soft sensor development and applications based on lstm in deep neural networks. In: IEEE. *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. [S.I.], 2017. p. 1–6.
- KIM, J.; EL-KHAMY, M.; LEE, J. Residual lstm: Design of a deep recurrent architecture for distant speech recognition. *arXiv preprint arXiv:1701.03360*, 2017.
- KORENIUS, T. et al. Stemming and lemmatization in the clustering of finnish text documents. In: *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. [S.I.: s.n.], 2004. p. 625–633.
- KRÖSE, B. et al. An introduction to neural networks. Citeseer, 1993.
- KUMAR, J.; GOOMER, R.; SINGH, A. K. Long short term memory recurrent neural network (lstm-rnn) based workload forecasting model for cloud datacenters. *Procedia Computer Science*, v. 125, p. 676–682, 2018. ISSN 1877-0509. The 6th International Conference on Smart Computing and Communications. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S1877050917328557>>.

- LAI, S. et al. Recurrent convolutional neural networks for text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, v. 29, n. 1, Feb. 2015. Disponível em: <<https://ojs.aaai.org/index.php/AAAI/article/view/951>>.
- LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. *Neural computation*, MIT Press, v. 1, n. 4, p. 541–551, 1989.
- LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Ieee, v. 86, n. 11, p. 2278–2324, 1998.
- LIDDY, E. D. Natural language processing. In: _____. *Encyclopedia of Library and Information Science*, 2nd. [S.l.: s.n.], 2001.
- LITTLE, W. A. The existence of persistent states in the brain. *Mathematical biosciences*, Elsevier, v. 19, n. 1-2, p. 101–120, 1974.
- LORIA, S. *TextBlob: Simplified Text Processing*. 2018. Disponível em: <<https://textblob.readthedocs.io/en/dev/>>.
- LOUIS, A. *A Brief History of Natural Language Processing*. 2020. <<https://medium.com/@antoine.louis/a-brief-history-of-natural-language-processing-part-1-ffbc937ebce>> e <<https://medium.com/@antoine.louis/a-brief-history-of-natural-language-processing-part-2-f5e575e8e37>>. Acesso em: 15 abr. 2020.
- LU, T. et al. Discriminative metric learning for face verification using enhanced siamese neural network. *Multimedia Tools and Applications*, Springer, v. 80, n. 6, p. 8563–8580, 2021.
- LUCCA, J. D.; NUNES, M. d. G. V. Lematização versus stemming. *USP, UFSCar, UNESP, São Carlos, São Paulo*, 2002.
- MAKREHCHI, M.; KAMEL, M. S. Automatic extraction of domain-specific stopwords from labeled documents. In: SPRINGER. *European Conference on Information Retrieval*. [S.l.], 2008. p. 222–233.
- MARCHISIO, A. et al. *A Methodology for Automatic Selection of Activation Functions to Design Hybrid Deep Neural Networks*. 2018.
- MAYBURY, M. *Advances in automatic text summarization*. [S.l.]: MIT press, 1999.
- MCCALLUM, A.; NIGAM, K. et al. A comparison of event models for naive bayes text classification. In: CITESEER. *AAAI-98 workshop on learning for text categorization*. [S.l.], 1998. v. 752, n. 1, p. 41–48.
- MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943.
- MELEKHOV, I.; KANNALA, J.; RAHTU, E. Siamese network features for image matching. In: IEEE. *2016 23rd International Conference on Pattern Recognition (ICPR)*. [S.l.], 2016. p. 378–383.
- MEMON, A. R. Similarity and plagiarism in scholarly journal submissions: bringing clarity to the concept for authors, reviewers and editors. *Journal of Korean medical science*, The Korean Academy of Medical Sciences, v. 35, n. 27, 2020.

- MEZIC, A. *Why Unsupervised Machine Learning is the Future of Cybersecurity*. 2020. Disponível em: <<https://technative.io/why-unsupervised-machine-learning-is-the-future-of-cybersecurity/>>. Acesso em: 01 mai. 2021.
- MIKOLOV, T. et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- MOLNAR, C. *Interpretable machine learning*. [S.l.]: Lulu. com, 2020.
- MOURA, W. *Word Embedding*. 2018. Disponível em: <<https://hackinganalytics.com/2018/01/31/word-embedding/>>. Acesso em: 19 jun. 2021.
- MUELLER, J.; THYAGARAJAN, A. Siamese recurrent architectures for learning sentence similarity. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2016. v. 30, n. 1.
- NECULOIU, P.; VERSTEEGH, M.; ROTARU, M. Learning text similarity with siamese recurrent networks. In: *Proceedings of the 1st Workshop on Representation Learning for NLP*. [S.l.: s.n.], 2016. p. 148–157.
- NICHOLSON, C. *Redes Neurais Articiais*. 2020. Disponível em: <<https://wiki.pathmind.com/neural-network>>. Acesso em: 30 abr. 2021.
- NOWLAN, S. J.; PLATT, J. C. A convolutional neural network hand tracker. *Advances in neural information processing systems*, Citeseer, p. 901–908, 1995.
- OLAH, C. *Understanding LSTM Networks*. 2015. Disponível em: <<https://colah.github.io/posts/2015-08-Understanding-LSTMs/>>. Acesso em: 10 mai. 2021.
- PAI, A. *CNN vs. RNN vs. ANN – Analyzing 3 Types of Neural Networks in Deep Learning*. 2020. Disponível em: <<https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/>>. Acesso em: 05 mai. 2021.
- PANETTO, H. et al. *On the Move to Meaningful Internet Systems: OTM 2019 Conferences: CoopIS, ODBASE, C&TC 2019, Proceedings*. [S.l.]: Springer, 2019. 760 p.
- PAWADE, D. et al. Story scrambler-automatic text generation using word level rnn-lstm. *International Journal of Information Technology and Computer Science (IJITCS)*, v. 10, n. 6, p. 44–53, 2018.
- PETEEL. *Redes Neurais e DeepLearning*. 2021. Disponível em: <<http://www.peteel.ufsc.br/2020/12/07/vies-inconsciente-machine-learning-2/>>. Acesso em: 26 mai. 2021.
- PHI, M. *Illustrated Guide to LSTM's and GRU's: A step by step explanation*. 2018. Disponível em: <<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>>. Acesso em: 11 mai. 2021.
- PIERCE, J. et al. Computers in translation and linguistics (alpac report). report 1416. *National Academy of Sciences/National Research Council*, 1966.
- PLISSON, J. et al. A rule based approach to word lemmatization. In: *Proceedings of IS*. [S.l.: s.n.], 2004. v. 3, p. 83–86.

- PONTES, E. L. et al. Predicting the semantic textual similarity with siamese cnn and lstm. *arXiv preprint arXiv:1810.10641*, 2018.
- PRADHAN, N.; GYANCHANDANI, M.; WADHVANI, R. A review on text similarity technique used in ir and its application. *International Journal of Computer Applications*, v. 120, p. 29–34, 06 2015.
- RADEV, D. R. et al. Mead-a platform for multidocument multilingual text summarization. 2004.
- RADFORD, A. et al. Language models are unsupervised multitask learners. *OpenAI blog*, v. 1, n. 8, p. 9, 2019.
- RADHAKRISHNAN, P. *What are Hyperparameters? and How to tune the Hyperparameters in a Deep Neural Network?* 2017. Disponível em: <<https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917>>. Acesso em: 14 jun. 2021.
- RAHGOZAR, A.; INKPEN, D. Semantics and homothetic clustering of hafez poetry. In: *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*. [S.l.: s.n.], 2019. p. 82–90.
- RANASINGHE, T.; ORASAN, C.; MITKOV, R. Semantic textual similarity with siamese neural networks. In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. [S.l.: s.n.], 2019. p. 1004–1011.
- RAVAL, S. *Generating Text using an LSTM Network (No libraries)*. 2017. Disponível em: <https://github.com/llSourcell/LSTM_Networks/blob/master/LSTM%20Demo.ipynb>. Acesso em: 12 mai. 2021.
- REN, B. The use of machine translation algorithm based on residual and lstm neural network in translation teaching. *Plos one*, Public Library of Science San Francisco, CA USA, v. 15, n. 11, p. e0240663, 2020.
- RENAUX, C. *Machine Learning: O que é, para que serve e exemplos práticos*. 2018. Disponível em: <<https://camilarenaux.com.br/videos/machine-learning-o-que-e-para-que-servir-e-exemplos-praticos/>>. Acesso em: 08 jun. 2021.
- ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958.
- RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, v. 323, n. 6088, p. 533–536, Oct 1986. ISSN 1476-4687. Disponível em: <<https://doi.org/10.1038/323533a0>>.
- SAIF, H. et al. On stopwords, filtering and data sparsity for sentiment analysis of twitter. 2014.
- SAK, H.; SENIOR, A. W.; BEAUFAYS, F. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. 2014.
- SANTHANAM, S. Context based text-generation using lstm networks. *arXiv preprint arXiv:2005.00048*, 2020.

- SARKAR, M.; BRUYN, A. D. Lstm response models for direct marketing analytics: Replacing feature engineering with deep learning. *Journal of Interactive Marketing*, Elsevier, v. 53, p. 80–95, 2021.
- SAS. *Deep Learning, O que é e qual sua importância?* 2021. Disponível em: <https://www.sas.com/pt_br/insights/analytics/deep-learning.html>. Acesso em: 04 mai. 2021.
- SCHANK, R. C. Conceptual dependency: A theory of natural language understanding. *Cognitive psychology*, Elsevier, v. 3, n. 4, p. 552–631, 1972.
- SHAHRIARI, B. et al. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, v. 104, n. 1, p. 148–175, 2016.
- SHANG, W.; ZHANG, J.; HUANG, W.-b. *Modelling Poetic Similarity: A Comparative Study of W. B. Yeats and the English Romantic Poets*. DataverseNL, 2019. Disponível em: <<https://doi.org/10.34894/OUOSLM>>.
- SHARMA, A. *Understanding Activation Functions in Neural Networks*. 2017. Disponível em: <<https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0>>. Acesso em: 08 set. 2021.
- SHARMA, P. *4 Types of Distance Metrics in Machine Learning*. 2020. Disponível em: <<https://www.analyticsvidhya.com/blog/2020/02/4-types-of-distance-metrics-in-machine-learning/>>. Acesso em: 26 ago. 2021.
- SHARMA, P. *Keras Dense Layer Explained for Beginners*. 2020. Disponível em: <<https://machinelearningknowledge.ai/keras-dense-layer-explained-for-beginners/>>. Acesso em: 08 set. 2021.
- SHEWALKAR, A. Performance evaluation of deep neural networks applied to speech recognition: Rnn, lstm and gru. *Journal of Artificial Intelligence and Soft Computing Research*, v. 9, n. 4, p. 235–245, 2019.
- SHORFUZZAMAN, M.; HOSSAIN, M. S. Metacovid: A siamese neural network framework with contrastive loss for n-shot diagnosis of covid-19 patients. *Pattern Recognition*, Elsevier, v. 113, p. 107700, 2021.
- SIEG, A. *Text Similarities: Estimate the degree of similarity between two texts*. 2018. Disponível em: <<https://medium.com/@adriensieg/text-similarities-da019229c894>>. Acesso em: 06 jun. 2020.
- SILVERMAN, B. W.; JONES, M. C. E. fix and j.l. hedges (1951): An important contribution to nonparametric discriminant analysis and density estimation: Commentary on fix and hedges (1951). *International Statistical Review / Revue Internationale de Statistique*, [Wiley, International Statistical Institute (ISI)], v. 57, n. 3, p. 233–238, 1989. ISSN 03067734, 17515823. Disponível em: <<http://www.jstor.org/stable/1403796>>.
- SOH, M. Learning cnn-lstm architectures for image caption generation. *Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, Tech. Rep.*, 2016.

- SOUTNER, D.; MÜLLER, L. Application of lstm neural networks in language modelling. In: SPRINGER. *International Conference on Text, Speech and Dialogue*. [S.l.], 2013. p. 105–112.
- SOUZA, A. de et al. Aplicação de redes neurais siamesas na autenticação de condutores. In: . [S.l.: s.n.], 2019.
- SRIVASTAVA, N. Improving neural networks with dropout. *University of Toronto*, v. 182, n. 566, p. 7, 2013.
- SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, JMLR.org, v. 15, n. 1, p. 1929–1958, 2014.
- STEINHAUS, H. et al. Sur la division des corps matériels en parties. *Bull. Acad. Polon. Sci*, v. 1, n. 804, p. 801, 1956.
- STRACHNYI, K. *Brief History of Neural Networks*. 2019. Disponível em: <<https://medium.com/analytics-vidhya/brief-history-of-neural-networks-44c2bf72eec>>. Acesso em: 01 mai. 2021.
- SU, M.-H. et al. A chatbot using lstm-based multi-layer embedding for elderly care. In: IEEE. *2017 International Conference on Orange Technologies (ICOT)*. [S.l.], 2017. p. 70–74.
- SUNDERMEYER, M.; NEY, H.; SCHLÜTER, R. From feedforward to recurrent lstm neural networks for language modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, IEEE, v. 23, n. 3, p. 517–529, 2015.
- SUNDERMEYER, M.; SCHLÜTER, R.; NEY, H. Lstm neural networks for language modeling. In: *Thirteenth annual conference of the international speech communication association*. [S.l.: s.n.], 2012.
- SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, 2014.
- TAN, Y. H.; CHAN, C. S. Phrase-based image caption generator with hierarchical lstm network. *Neurocomputing*, Elsevier, v. 333, p. 86–100, 2019.
- TEJA, S. *Stop Words in NLP*. 2020. Disponível em: <<https://medium.com/@saitejaponugoti/stop-words-in-nlp-5b248dadad47>>. Acesso em: 05 jun. 2021.
- TEJADA, Z. et al. *Working with CSV and JSON files for data solutions*. 2018. Disponível em: <<https://docs.microsoft.com/en-us/azure/architecture/data-guide/scenarios/csv-and-json>>. Acesso em: 08 jun. 2021.
- TOMALOK, E. *Word2Vec e sua importância na etapa de pré-processamento*. 2018. Disponível em: <<https://medium.com/@everton.tomalok/word2vec-e-sua-import%C3%A1ncia-na-etapa-de-pr%C3%A9-processamento-d0813acfc8ab>>. Acesso em: 19 jun. 2021.
- TOPRAK, M. *Save it! Do Not Waste Your Time*. 2019. Disponível em: <<https://medium.com/@toprak.mhmt/save-it-do-not-waste-your-time-d9b461350f83>>. Acesso em: 08 jun. 2021.

- VAZ, A. L. *Introdução teórica a Neural Network — Deep Learning — Parte 1.* 2018. Disponível em: <<https://medium.com/data-hackers/neural-network-deep-learning-parte-1-introdu%C3%A7%C3%A3o-te%C3%B3rica-5c6dcd2e5a79>>. Acesso em: 29 abr. 2021.
- VAZ, A. L. *Otimizando os hiperparâmetros.* 2019. Disponível em: <<https://medium.com/data-hackers/otimizando-os-hiperpar%C3%A2metros-621de5e9be37>>. Acesso em: 09 set. 2021.
- VIJAYMEENA, M.; KAVITHA, K. A survey on similarity measures in text mining. *Machine Learning and Applications: An International Journal*, v. 3, n. 2, p. 19–28, 2016.
- VÁZQUEZ, F. *Deep Learning made easy with Deep Cognition.* 2017. Disponível em: <<https://becominghuman.ai/deep-learning-made-easy-with-deep-cognition-403fbe445351>>. Acesso em: 05 mai. 2021.
- WIDROW, B.; LEHR, M. A. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, IEEE, v. 78, n. 9, p. 1415–1442, 1990.
- WIGGERS, K. L. et al. Image retrieval and pattern spotting using siamese neural network. In: IEEE. *2019 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2019. p. 1–8.
- WILLIAMS, R. J.; ZIPSER, D. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, MIT Press, v. 1, n. 2, p. 270–280, 1989.
- WOLFF, R. *11 Natural Language Processing (NLP) Applications in Business.* 2020. Disponível em: <<https://monkeylearn.com/blog/natural-language-processing-applications/>>. Acesso em: 21 abr. 2021.
- WOODS, W. A. Transition network grammars for natural language analysis. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 13, n. 10, p. 591–606, out. 1970. ISSN 0001-0782. Disponível em: <<https://doi.org/10.1145/355598.362773>>.
- WU, J. et al. Hyperparameter optimization for machine learning models based on bayesian optimizationb. *Journal of Electronic Science and Technology*, v. 17, n. 1, p. 26–40, 2019. ISSN 1674-862X. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S1674862X19300047>>.
- XU, A. et al. A new chatbot for customer service on social media. In: *Proceedings of the 2017 CHI conference on human factors in computing systems*. [S.l.: s.n.], 2017. p. 3506–3510.
- YIH, W.-t. et al. Learning discriminative projections for text similarity measures. In: *Proceedings of the fifteenth conference on computational natural language learning*. [S.l.: s.n.], 2011. p. 247–256.
- ZHANG, L.; GAO, J. et al. A comparative study to understanding about poetics based on natural language processing. *Open Journal of Modern Linguistics*, Scientific Research Publishing, v. 7, n. 05, p. 229, 2017.
- ZHENG, R. et al. A framework for authorship identification of online messages: Writing-style features and classification techniques. *Journal of the American society for information science and technology*, Wiley Online Library, v. 57, n. 3, p. 378–393, 2006.

ZORNOZA, J. *Distance Metrics for Machine Learning*. 2020. Disponível em: <<https://aigents.co/blog/publication/distance-metrics-for-machine-learning>>. Acesso em: 19 set. 2021.