

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E INOVAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Francisco Daniel Coutinho Medeiros

**IDALina - Uma proposta (I)nteligente de aprendizagem profunda
para (D)eteccção (A)utomática de notícias falsas na (Lín)gua
Portugues(a)**

Fortaleza – CE

9 de abril de 2021

Francisco Daniel Coutinho Medeiros

**IDALina - Uma proposta (I)nteligente de aprendizagem
profunda para (D)eteccção (A)utomática de notícias
falsas na (Lín)gua Portugues(a)**

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Linha de Pesquisa: Inteligência Artificial

Orientador: Prof. Dr. Reinaldo Bezerra Braga

Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)

Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Fortaleza – CE

9 de abril de 2021

Dados Internacionais de Catalogação na Publicação
Instituto Federal do Ceará - IFCE
Sistema de Bibliotecas - SIBI
Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

M488i Medeiros, Francisco Daniel Coutinho.

IDALina - Uma proposta (I)nteligente de aprendizagem profunda para (D)etecção (A)utomática de notícias falsas na (Lín)gua Portuguesa / Francisco Daniel Coutinho Medeiros. - 2021.
120 f. : il. color.

Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Ciência da Computação, Campus Fortaleza, 2021.

Orientação: Prof. Dr. Reinaldo Bezerra Braga.

1. Inteligência Artificial. 2. Redes Neurais. 3. Aprendizagem Profunda. 4. Processamento de Linguagem Natural. I. Título.

CDD 004



Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)
Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)
Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Francisco Daniel Coutinho Medeiros

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, sendo aprovada pela Coordenação do Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará e pela banca examinadora:

Orientador: Prof. Dr. Reinaldo Bezerra Braga
Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Profa. Dra. Carina Teixeira de Oliveira
Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Prof. Dr. Lincoln Souza Rocha
Universidade Federal do Ceará (UFC)

Fortaleza – CE
9 de abril de 2021

Dedico este trabalho à minha querida mãe Raimunda (in memoriam), cujo empenho em me educar sempre veio em primeiro lugar. Aqui estão os resultados dos seus esforços, mãe.

Muita gratidão e saudade.

Agradecimentos

Agradeço, primeiramente, à Deus por me proporcionar perseverança, não só durante este trabalho, mas durante toda a minha vida.

Agradeço à minha mãe Raimunda (in memoriam), por tudo que sou e sei que lá de cima deve estar orgulhosa da trajetória do filho.

Agradeço ao meu pai Marcelo, por ter me ensinado valores que carrego comigo em todos os momentos.

Agradeço, com muito carinho, à minha esposa Idalina. Meu amor, sem seu apoio e companheirismo, tudo isso não seria possível. Obrigado por sempre acreditar em mim e entender minha ausência nos momentos mais críticos deste trabalho.

Agradeço à minha sogra Maria Valdice, que é como uma mãe para mim e sempre esteve ao meu lado me apoiando ao longo de toda a minha caminhada.

Em especial, agradeço ao meu orientador Prof. Reinaldo Bezerra Braga, por sua confiança e otimismo. Você nunca perdeu a fé na minha pesquisa e sempre soube me tranquilizar nos momentos mais difíceis. Obrigado por sempre me manter motivado durante todo este árduo processo.

Aos membros da banca examinadora, Profa. Carina Teixeira de Oliveira e Prof. Lincoln Souza Rocha, que, mesmo com o curto prazo para leitura, no qual peço sinceras desculpas, tão gentilmente aceitaram participar e colaborar com esta dissertação.

Agradeço a todos os meus familiares pelo apoio e incentivo em todos os momentos e aos meus amigos e colegas, pelas trocas de conhecimentos, conselhos e incentivos.

E agradeço a todas as pessoas que de alguma forma colaboraram para que este trabalho fosse realizado.

“As convicções são inimigas mais perigosas da verdade do que as mentiras.”
(Friedrich Nietzsche)

Resumo

O crescimento das redes sociais alavancou o consumo de notícias por conta de seu fácil acesso e baixo custo dos veículos de comunicação. Contudo, com o advento das redes sociais, a Internet se tornou o ambiente perfeito para a disseminação de notícias falsas, conhecidas popularmente como *Fake News*. No contexto mundial, onde as informações são transmitidas e compartilhadas rapidamente, torna-se cada vez mais evidente a necessidade de prezar pela credibilidade e idoneidade do veículo transmissor de conteúdo. Portanto, esta dissertação se coloca como sendo um importante passo no sentido de alavancar a pesquisa nacional na área de detecção de notícias falsas em português e caminhar na direção do aumento de confiabilidade dos veículos de imprensa nacionais. Inicialmente, por meio de uma revisão sistemática, a primeira contribuição deste trabalho está relacionada à identificação da pesquisa científica em torno da detecção de notícias falsas em português, que ainda encontra-se em um estado incipiente. Esta lacuna motivou a criação da IDALina - Uma proposta (I)nteligente de aprendizagem profunda para (D)etecção (A)utomática de notícias falsas na (Lín)gua Portugues(a). IDALina propõe uma arquitetura híbrida, composta por redes convolucionais e recorrentes, que visa classificar notícias falsas por meio de suas características semânticas, morfosintáticas e sentimentais obtidas através de técnicas de processamento de linguagem natural. Após um conjunto de experimentos preliminares, que otimizaram as medidas de performance da arquitetura proposta, IDALina foi comparada com várias arquiteturas distintas e propostas com soluções consideradas estado-da-arte, obtendo uma acurácia de 98,1%, com uma taxa de falso positivo de 1,9%. Desta forma, concluiu-se que a solução proposta supera as medidas de performance em comparação com todos estes modelos utilizados até então para resolver o problema de notícias falsas em português. Contudo, a principal contribuição desta dissertação está na alavancagem da discussão científica e da pesquisa nacional acerca do tema. Além do impacto pela sua natureza inovadora, há também o impacto social deste estudo de propor ferramentas e soluções que mitiguem o problema da proliferação de notícias falsas em português.

Palavras-chave: Inteligência Artificial. Redes Neurais. Aprendizagem Profunda. Redes Convolucionais. Redes Recorrentes. Processamento de Linguagem Natural.

Abstract

The growth of social networks platforms leverages the consumption of news due to its easy access, spreading behavior, and low cost. However, this revolution in the way that information is released has provided the growth of something that always walked side by side with the real news: fake news. From a global perspective, where information is received and shared faster, it is increasingly evident that it is necessary to enrich the news credibility of the mainstream news vehicles. Therefore, this proposal stands an important step towards leveraging national research in the area of fake news detection in Portuguese and moving forward on increasing the reliability of brazilian mainstream vehicles. We identified, through a systematic review that is one of the contributions of this work, that the scientific research around the detection of fake news in Portuguese is still incipient. This gap motivated the creation of IDALina - from Portuguese “*Uma proposta (I)nteligente de aprendizagem profunda para (D)etecção (A)utomática de notícias falsas na (Lín)gua Portugues(a)*”. IDALina adopts a hybrid architecture, composed by Convolutional and Recurrent Neural Networks, which aims to detect fake news using semantic, morphosyntactic and sentimental features obtained by Natural Language Processing tasks. After some fine-tuning experiments, which aims to optimize the proposed architecture, IDALina was compared with several other architectures and with state-of-the-art studies. The proposed network obtained 98.1% accuracy, and 1.9% of false positive rate. This results outperforms the state-of-the-art solutions. We expect that this work leverages the national research about this topic. Besides of its innovative nature, this study also aims impact socially by proposing tools and solutions that mitigates the problem of spreading of fake news in Portuguese.

Keywords: Artificial Intellingence, Neural Networks, Deep Learning, Convolutional Networks, Recurrent Networks, Natural Language Processing.

Lista de ilustrações

Figura 1 – Diferentes abordagens de detecção de notícias falsas (adaptado de (BONDIELLI; MARCELLONI, 2019a))	33
Figura 2 – Etiquetagem segundo o método Brill (BRILL, 1994)	36
Figura 3 – Representação gráfica do modelo VSM (adaptado de (ZHAI; MASSUNG, 2016))	42
Figura 4 – Word2Vec: palavras que são semanticamente semelhantes estão próximas no espaço vetorial (Fonte: (AMAZON, 2020))	45
Figura 5 – Similaridade entre vetores que se relacionam em um contexto semântico, chamado gênero (vetor pontilhado)	45
Figura 6 – Modelo CBoW (RONG, 2014)	46
Figura 7 – Modelo Skip-Gram (RONG, 2014)	48
Figura 8 – RNA com duas camadas escondidas (Fonte: Deng e Liu (2018))	52
Figura 9 – Representação de um Neurônio Artificial (fonte: autor)	52
Figura 10 – Convolução 1D, com filtro $k = 3$ (fonte: autor)	55
Figura 11 – Operação do <i>Max-Pooling</i> (fonte: autor)	56
Figura 12 – Estrutura de uma RNN com modelo de realimentação Many-to-One (esquerda) e Many-to-Many (direita) (adaptado de (HAYKIN; NETWORK, 2004))	57
Figura 13 – Representação de uma célula LSTM (adaptado de (HOCHREITER; SCHMIDHUBER, 1997))	58
Figura 14 – Diagrama PRISMA adaptado para este trabalho.	63
Figura 15 – Quantidade de estudos por ano.	64
Figura 16 – Algoritmos de IA mais utilizados.	67
Figura 17 – Combinações de conteúdo consideradas.	68
Figura 18 – IDALina.	72
Figura 19 – Diagrama de caixa para o tamanho dos textos do <i>dataset</i>	82
Figura 20 – Acurácia dos otimizadores ao longo do treinamento.	84
Figura 21 – Impacto do tamanho do lote na performance do modelo.	86
Figura 22 – Performance do Modelo Proposto (cor roxa) em relação à demais RNAs	89
Figura 23 – Matriz de confusão da IDALina para dados fora da base de treinamento e validação.	92
Figura 24 – Fragmento da taxonomia WordNet. Linhas sólidas representam ligações "é um" e linhas tracejadas indicam que nós foram omitidos. (RESNIK, 1995)	114
Figura 25 – Técnica de <i>dropout</i> . Á rede completamente conectada da esquerda é submetida a um <i>dropout</i> , resultando na rede da figura à direita.	118

Lista de tabelas

Tabela 1	–	<i>Tagset</i> do corpus MAC-Morpho	38
Tabela 2	–	<i>Tagset</i> do corpus LX-Corpus	40
Tabela 3	–	Tabela PICO (<i>Population, Intervention, Comparison, and Outcomes</i>) para estruturação do Protocolo de Busca	60
Tabela 4	–	Protocolo de busca utilizados nos motores de busca	61
Tabela 5	–	Quantidade de publicações por tipo.	65
Tabela 6	–	Resultados de detecção dos artigos selecionados	66
Tabela 7	–	Análise do Corpus Fake.Br (MONTEIRO; SANTOS; B, 2018)	78
Tabela 8	–	Performance de classificação considerando apenas as entradas dos eti- quetadores	80
Tabela 9	–	Comparação entre os resultados do modelo proposto e as demais abor- dagens.	90
Tabela 10	–	Comparativo das contribuições desta dissertação com os trabalhos relacionados.	91

Lista de abreviaturas e siglas

NLP	<i>Natural Language Processing</i>
RNA	Rede Neural Artificial
RNP	Rede Neural Profunda
CNN	<i>Convolutional Neural Network</i>
RNN	<i>Recurrent Neural Network</i>
LSTM	<i>Long Sort-Term Memory</i>
Bi-LSTM	<i>Bidirectional Long Sort-Term Memory</i>
GRU	<i>Gated recurrent unit</i>
MLP	<i>Multi-Layer Perceptron</i>
SVM	<i>Support Vector Machine</i>
POS-Tag	<i>Parts-of-Speech Tagging</i>
TBL	<i>Transformation-Based Learning</i>
SRL	<i>Semantic Role Labeling</i>
BoW	<i>Bag-of- Words</i>
VSM	<i>Vector Space Model</i>
IR	<i>Information Retrieval</i>
TF-IDF	<i>Term Frequency-Inverse Document Frequency</i>
LSA	<i>Latent Semantic Analysis</i>
CBoW	<i>Continuous Bag-of- Words</i>
SVD	<i>Singular Value Decomposition</i>
NNLM	<i>Neural Network Language Model</i>
GloVe	<i>Global Vectors for Word Representation</i>
LIWC	<i>Linguistic Inquiry and Word Count</i>

ReLU	<i>Rectified Linear Unit</i>
ACM	<i>Association for Computing Machinery</i>
RS	Revisão Sistemática
HAN	<i>Hierarchical Attention Network</i>
LPT	<i>Labeled Propagation Tree</i>
LR	<i>Logistic Regression</i>
HBLC	<i>Harmonic Boolean Label Crowdsourcing</i>
NMF	<i>Non-negative Matrix Factorization</i>
CRF	<i>Conditional Random Fields</i>
NLTK	<i>Natural Language Toolkit</i>
SGD	<i>Stochastic Gradient Descent</i>

Sumário

1	INTRODUÇÃO	25
1.1	Objetivos Geral e Específicos	26
1.2	Estrutura da Dissertação	28
2	FUNDAMENTAÇÃO TEÓRICA	29
2.1	Contextualizando as Fake News	29
2.1.1	Ecosistema das Fake News	29
2.1.1.1	Tipos de Fake News	29
2.1.2	Aspectos Psicológicos	31
2.1.3	Técnicas de Detecção	32
2.2	Processamento de Linguagem Natural	33
2.2.1	Técnicas de Pré-Processamento	34
2.2.1.1	Corpus	34
2.2.1.2	Filtragem, Tokenização, Lematização e Radicalização	34
2.2.1.3	Stopwords	35
2.3	Parts-of-Speech Tagging	35
2.3.1	Brill Tagger	36
2.3.1.1	Corpus MAC-Morpho	37
2.3.2	NLPNet	37
2.3.3	LX-Tagger	39
2.3.3.1	LX-Corpus	39
2.3.4	HunPos	39
2.4	Modelos de Representação Textual	41
2.4.1	Representação Vetorial de Documentos	41
2.4.1.1	One-Hot Encoding	42
2.4.1.2	Term Frequency-Inverse Document Frequency (TF-IDF)	42
2.4.2	Representação Distribuída das Palavras	42
2.4.2.1	Latent Semantic Analysis (LSA)	43
2.4.2.2	Singular Value Decomposition (SVD)	43
2.4.3	Métodos Preditivos	44
2.4.3.1	Word2Vec	44
2.4.3.1.1	Continuous Bag-of-Words (CBoW)	46
2.4.3.1.2	Skip-Gram	48
2.4.3.1.3	CBoW vs. Skip-Gram	49
2.4.4	GloVe	49

2.5	Linguistic Inquiry and Word Count (LIWC)	50
2.6	Redes Neurais Artificiais	51
2.6.1	Neurônios Artificiais	51
2.6.1.1	Funções de Ativação	52
2.6.2	Perceptron Multicamadas	53
2.7	Aprendizagem Profunda	53
2.7.1	Redes Neurais Convolucionais	54
2.7.1.1	Convolução 1D	54
2.7.1.2	Vetor de Pooling 1D	55
2.7.2	Redes Neurais Recorrentes	56
2.7.3	Long Short-Term Memory (LSTM)	57
3	REVISÃO SISTEMÁTICA	59
3.1	Planejamento	59
3.1.1	Questões de Pesquisa	59
3.1.2	Protocolo de Busca	59
3.1.3	Critérios de Inclusão/Exclusão	60
3.1.3.1	Critérios de Inclusão	61
3.1.3.2	Critérios de Exclusão	61
3.1.3.3	Critérios de Qualidade	62
3.2	Condução	62
3.3	Resultados da RS	64
3.3.1	RQ1: Como estão distribuídas as publicações sobre detecção de notícias falsas ao longo dos anos?	64
3.3.2	RQ2: Quais tipos de contribuições têm sido desenvolvidos no contexto da detecção de notícias falsas nas redes sociais?	64
3.3.3	RQ3: Quais são os resultados de detecção dos artigos mais qualificados?	65
3.3.4	RQ4: Quais algoritmos de Inteligência Artificial são aplicados nos estudos selecionados?	67
3.3.5	RQ5: Quais tipos de conteúdo são considerados para as soluções propostas?	68
3.4	Ameaças à Validade	68
3.5	Conclusão da RS	68
4	IDALINA	71
4.0.1	Camada de Entrada	71
4.0.2	Camada de <i>Embeddings</i>	73
4.0.3	Camadas CNN	73
4.0.4	Camada de Concatenação	73
4.0.5	Camada Recorrente Bidirecional	74
4.0.6	Camada Densamente Conectada	74

4.0.7	Camada de Saída	75
5	EXPERIMENTOS E ANÁLISES	77
5.1	Escolha do <i>dataset</i>	77
5.1.1	Análise do Corpus	77
5.2	Desenvolvimento e Ferramentas Adotadas	78
5.2.1	<i>Natural Language Toolkit</i> (NLTK)	78
5.2.2	Keras e Tensorflow	78
5.3	Escolha do Etiquetador	79
5.4	Ajuste de Hiperparâmetros	80
5.4.1	Tamanho do Texto	81
5.4.2	<i>Learning Rate</i>	83
5.4.2.1	Escolha do Otimizador	83
5.4.3	Tamanho do Lote	85
5.4.4	Demais Parâmetros	86
5.5	Resultados	87
5.5.1	Processo de Treinamento, Validação e Teste	87
5.5.2	IDALina <i>versus</i> RNAs Clássicas	88
5.5.3	IDALina <i>versus</i> Demais Abordagens	88
5.5.3.1	Medidas de Performance	89
5.5.3.2	Contribuições deste Trabalho	90
5.5.4	Desempenho do Modelo fora da Base de Treinamento e Validação	91
6	CONCLUSÃO	93
6.1	Trabalhos Futuros	93
6.2	Considerações Finais	94
	REFERÊNCIAS	97
	APÊNDICES	109
	APÊNDICE A – MEDIDAS DE ASSOCIAÇÃO	111
A.1	Matriz Palavra-Contexto	111
A.1.0.1	Pointwise Mutual Information (PMI)	111
	APÊNDICE B – MEDIDAS DE SIMILARIDADE	113
B.1	Informational Content	113
B.2	Similaridade de Resnik	113
B.3	Similaridade de Lin	113

B.4	Cosseno	114
-----	-------------------	-----

ANEXOS 115

ANEXO A – TÉCNICAS RELACIONADOS A *DEEP LEARNING* . 117

A.1	Gradiente Descendente Estocástico	117
A.1.1	Batch Normalization	117
A.1.2	Dropout	118

1 Introdução

O crescimento das redes sociais alavancou o consumo de notícias por conta de seu fácil acesso e baixo custo dos veículos de comunicação. Contudo, com o advento das redes sociais, a Internet se tornou o ambiente perfeito para a disseminação de notícias falsas, conhecidas popularmente como *Fake News*. Essas notícias têm gerado um significativo impacto na vida diária da sociedade, causando fortes danos econômicos às empresas e, até mesmo, influenciando direta ou indiretamente decisões políticas de um país (LIU; WU, 2018). Portanto, este problema é visto como uma das grandes ameaças mundiais, principalmente quando relacionado à democracia, ao jornalismo e à liberdade de expressão. Ou seja, as *Fake News* enfraquecem instituições e influenciam decisões importantes na humanidade, tais como o referendo do “Brexit”, que foi um processo de saída do Reino Unido da União Européia, e as divididas eleições norte-americanas de 2016 (POGUE, 2017). Porém, não são apenas decisões políticas que as notícias falsas podem atacar. Isto significa que, nem mesmo o mercado financeiro mundial está livre deste problema. Um exemplo foi, em Abril de 2013, uma publicação de um texto falso afirmando que aconteceram duas explosões na Casa Branca e que o Presidente Barack Obama estava ferido. Este falso evento resultou em um prejuízo de mais de 200 bilhões de dólares em alguns minutos¹.

Como se já não bastassem os prejuízos causados por esse problema, o advento das redes sociais potencializaram o alcance das notícias falsas. Nestas redes sociais, os disseminadores (sejam eles humanos ou robôs) encontram um ambiente perfeito para disseminar conteúdos falsos. Em paralelo, as redes sociais contribuem para o surgimento de um fenômeno de grupo, chamado de *câmaras de eco ideológico*. Ou seja, os consumidores destes grupos são seletivamente expostos às notícias que possuem uma alta afinidade com suas ideologias (JAMIESON; CAPPELLA, 2008). Isto corrobora com o estudo de (TUFEKCI, 2015), onde o autor mostra que as pessoas nestas condições tendem a não clicar em matérias de assuntos que elas discordam.

Todavia, detectar notícias falsas automaticamente não é uma tarefa fácil, pois até mesmo humanos podem falhar em checar manualmente estas notícias (RUCHANSKY, 2017). Felizmente, os estudos mais proeminentes, acerca dessa linha de pesquisa, provaram que a combinação de Processamento de Linguagem Natural (do inglês *Natural Language Processing*, ou abreviadamente *NLP*) e algoritmos de aprendizagem de máquina (*Machine Learning*), em especial as Redes Neurais Artificiais (RNA), têm apresentado resultados expressivos para solucionar o problema da detecção automatizada de notícias falsas (OSHIKAWA; QIAN; WANG, 2018a; BONDIELLI; MARCELLONI, 2019b). Esses resultados

¹ <https://www.hindustantimes.com/business/fake-obama-injured-tweet-causes-fall-in-us-market/story-OqJZpBkqrCAW0hFKoEhqCM.html>

positivos norteiam a comunidade científica no estudo de estruturas de aprendizagem e, conseqüentemente, na geração de ferramentas que proporcionem uma boa detecção automatizada de notícias falsas a qualquer momento e em qualquer lugar.

Com o objetivo de identificar o estado da arte, bem como as oportunidades de contribuição científica, uma revisão sistemática aprofundada foi elaborada nesta temática, que resultou em artigo publicado em evento de abrangência nacional (MEDEIROS; BRAGA, 2020). Neste artigo, os autores concluem que as estruturas mais utilizadas para o problema são as RNAs, em particular as Redes Convolucionais (CNN) e as Redes Recorrentes de Memória Curta, do inglês *Long Sort-Term Memory* (LSTM). Destaca-se que os artigos mais relevantes acerca desta linha de pesquisa fazem uso de vários modelos de representação textual. Por esta razão, são propostas diversas arquiteturas para mitigar o problema da detecção automática de Fake News, obtendo resultados satisfatórios para o idioma inglês.

Entretanto, o trabalho de Medeiros e Braga (2020) mostra que há uma deficiência proeminente na pesquisa de detecção de notícias falsas para o idioma português. Desta forma, até onde se sabe, este é um dos primeiros trabalhos que propõe mitigar este problema por meio de uma arquitetura de Aprendizagem Profunda considerando aspectos morfológicos, sintáticos, semânticos e psicológicos (tais como, afetividade, raiva, certeza, dentre outros)

No contexto mundial, onde as informações são transmitidas e compartilhadas rapidamente, torna-se mais evidente a necessidade de prezar pela credibilidade e idoneidade do veículo transmissor de conteúdo. Portanto, esta proposta se coloca como sendo um importante passo no sentido de alavancar a pesquisa nacional na área de detecção de notícias falsas em português e caminhar na direção do aumento de confiabilidade dos veículos de imprensa brasileiros perante a sociedade.

1.1 Objetivos Geral e Específicos

O objetivo geral da presente dissertação consiste em propor uma arquitetura de detecção automática de notícias falsas em português na Web por meio de uma arquitetura de aprendizagem profunda.

Deste escopo geral, podemos elencar quatro objetivos específicos:

- OE1** Apresentar a revisão sistemática para identificar o estado da arte, bem como as oportunidades de contribuições que motivaram este trabalho;
- OE2** Propor uma arquitetura de aprendizagem profunda que obtenha melhores resultados que os obtidos com classificadores clássicos e outras RNAs mais simples. Além de superar os resultados dos trabalhos de Monteiro, Santos e B (2018) e Silva et al. (2020);

OE3 Avaliar o impacto das características morfológicas, sintáticas, semânticas e psicológicas no processo de detecção;

OE4 Alavancar a pesquisa nacional acerca do tema.

Por tratar-se de um tema que foi, de fato, popularizado durante as eleições americanas de 2016 (FOURNEY et al., 2017), é necessário um levantamento bibliográfico detalhado para identificar o estado da arte e as lacunas de pesquisa. Este levantamento obteve resultados que enfatizaram a característica inovadora e desafiadora desta proposta em face à escassez de trabalhos voltados para a língua portuguesa. Desta forma, **OE1** foi considerado um dos objetivos desta dissertação, dado que suas informações e conclusões, por si só, já representam uma importante contribuição científica.

A partir dos resultados da revisão sistemática, observou-se que os principais trabalhos nesta linha de pesquisa utilizavam estruturas de Aprendizagem Profunda para solucionar o problema. Além disso, concluiu-se em **OE1** que os poucos trabalhos encontrados em português utilizavam apenas classificadores clássicos, como *Support Vector Machines* (SVM), Naive Bayes e *Logistic Regression* (LR) como prova de conceito. Logo, pode-se inferir que resultados melhores podem ser obtidos a partir do uso de Redes Neurais Profundas (RNP), uma vez que estas estruturas são utilizadas para resolver problemas mais complexos, como o assunto em questão.

Desta forma, o objetivo de **OE2** consiste em definir uma arquitetura de rede profunda que resolva o problema de detecção de notícias falsas na Internet. A partir daí, originou-se a arquitetura proposta, IDALina, que consiste, resumidamente, de uma arquitetura híbrida composta por uma camada de convolução e outra recorrente. A camada de convolução se concentra em aprender as características locais e posicionais, ao passo que a camada recorrente é composta por uma rede LSTM bidirecional que aprende as características sequenciais do documento. IDALina foi comparada com várias arquiteturas distintas e trabalhos com soluções consideradas estado-da-arte, superando em todas as medidas de performance todos estes modelos apresentando uma acurácia de 98,1% com uma taxa de falso positivo de 1,9%. Ou seja, IDALina apresentou uma queda na taxa de falso positivo de, pelo menos, 25%.

Vale salientar que o *dataset* utilizado nos experimentos possui uma abrangência de notícias nos mais diversos temas, tais como política, saúde, economia, ciência e tecnologia, religião, entretenimento e celebridades. Logo, IDALina possui um amplo espectro de atuação.

O objetivo de **OE3** é descrever e avaliar o impacto das características morfológicas, sintáticas, semânticas e psicológicas no processo de detecção. Resumidamente, em NLP, as características morfossintáticas de um texto podem ser descritas matematicamente através de Etiquetadores Morfossintáticos (do inglês *Parts-of-Speech Tagging*, ou abreviadamente

POS-Tag). Esta proposta analisa os principais etiquetadores disponíveis para o idioma português nos experimentos preliminares.

As características semânticas fornecem ao modelo informações acerca do significado das palavras. Estas informações são obtidas por meio de algoritmos de aprendizagem pré-treinados, que possibilitam a representação vetorial de palavras no espaço semântico, auxiliando o modelo de predição com entradas comparações vetoriais de palavras. Finalmente, as características psicológicas tratam de representações mais subjetivas, tais como, afeto, raiva, certeza, emotividade, dentre outras.

Finalmente, o objetivo de **OE4** constitui-se como uma das principais contribuições deste trabalho. Afinal, é necessário estimular o debate científico e a pesquisa nacional acerca de propor soluções de impacto direto no público-alvo, que é a sociedade brasileira.

1.2 Estrutura da Dissertação

Esta dissertação está organizada da seguinte forma:

O Capítulo 2 trata da fundamentação teórica das técnicas utilizadas. Inicialmente, é feita uma contextualização do problema da proliferação de notícias falsas na Internet, detalhando suas particularidades e formas de detecção. Em seguida, são abordados conceitos referentes às técnicas de pré-processamento textual, modelos de representação textual, etiquetadores, redes neurais e aprendizagem profunda.

O Capítulo 3 descreve a revisão sistemática produzida como levantamento bibliográfico. Ela consiste de um robusto mapeamento de trabalhos relacionados que identifica o estado da arte e as oportunidades de pesquisa encontradas nessa linha de pesquisa, reforçando a revelância da contribuição científica contida na presente dissertação.

O Capítulo 4 apresenta a arquitetura proposta. Inicialmente é feita a modelagem do problema. Em seguida, a arquitetura proposta é discutida, justificando as camadas mais relevantes da estrutura.

O Capítulo 5 traz uma explanação dos experimentos realizados, bem como as ferramentas utilizadas. Cada abordagem é detalhada e confrontada com o modelo proposto. Ao final deste capítulo, é feita a discussão e análise dos resultados obtidos.

Por fim, no Capítulo 6 são elencadas as conclusões obtidas pela pesquisa e perspectivas de trabalhos futuros acerca do tema.

2 Fundamentação Teórica

Este capítulo apresenta os conceitos necessários para o desenvolvimento deste projeto. Por se tratar de uma linha de pesquisa relativamente nova, é preciso, inicialmente, contextualizar o problema da proliferação das *Fake News*, expondo suas características e formas de detecção. Em seguida, são discutidas as técnicas de Processamento de Linguagem Natural utilizadas neste trabalho, tais como pré-processamento, modelos de representação vetorial com palavras embutidas (*word embeddings*), etiquetagem morfosintática (*POS-Tagging*). Ao final, algumas estruturas de Redes Neurais Artificiais são apresentadas sob o aspecto da classificação de documentos.

2.1 Contextualizando as Fake News

As *Fake News* existem desde que as notícias começaram a circular ¹. Entretanto, existem várias definições para este termo. Allcott e Gentzkow (2017) definem “*Fake News*” como notícias que são intencionalmente e certificadamente falsas e que podem enganar o leitor. Esta definição se baseia em dois conceitos-chave: *autenticidade* e *intenção*. A autenticidade consiste nas simples informações erradas que estão contidas em uma notícia falsa. Por outro lado, as *Fake News* são criadas com a intenção de iludir o leitor.

Desta forma, faz-se necessário entender como funciona o ecossistema das notícias falsas para, então, poder identificar onde atuar no sentido de coibi-las.

2.1.1 Ecossistema das Fake News

Este ecossistema foi baseado na leitura do artigo escrito por Wardle (2017). Especificamente, os autores categorizam os vários tipos de notícias falsas que podem ser encontradas na Internet, os atores que contribuem para sua disseminação, bem como suas motivações. Entretanto, como o intuito deste capítulo é oferecer um contexto geral do problema como um todo afim de definir o escopo de atuação neste projeto, este ecossistema não deve ser considerado como uma representação detalhada e completa para todo o cenário da desinformação digital.

2.1.1.1 Tipos de Fake News

Notícias falsas podem ser encontradas de diversas formas. Sendo assim, elas podem ser categorizadas em sete grupos listados abaixo:

¹ <https://www.politico.com/magazine/story/2016/12/fake-news-history-long-violent-214535>

Propaganda: São notícias fabricadas para um determinado grupo e, normalmente, usadas em um contexto político. Este tipo de notícia falsa não é novidade, uma vez que ela se fez presente na 2ª Guerra Mundial e na Guerra Fria. Propagandas são usadas fortemente em contextos políticos, com o intuito de denegrir um partido opositor ou até mesmo uma nação em particular. No Brasil, este artifício é intensamente utilizado durante as campanhas eleitorais, persuadindo o eleitor contra o candidato opositor com notícias puramente falsas ou parcialmente verdadeiras, mas manipuladas para gerar confusão a quem consome a propaganda.

Teorias da Conspiração: São compostas por histórias que tentam explicar um acontecimento ou evento invocando uma conspiração sem provas. Geralmente, essas histórias são sobre atos executados por governos ou pessoas poderosas. Estas teorias costumam apresentar informações sem fonte como fatos ou dispensam inteiramente uma abordagem baseada em crenças ao invés de evidências.

Boatos: Notícias que contêm fatos completamente falsos e são divulgados como legítimos. Essa categoria também é conhecida na comunidade jornalística como factóides (DICKSON, 2014). Exemplos populares de boatos são histórias que relatam a falsa morte de celebridades (por exemplo, o boato da morte do ator Lima Duarte ²).

Rumores: Tratam-se de notícias cuja veracidade é ambígua ou nunca confirmada. Esse tipo de informação falsa é amplamente propagada nas redes sociais, portanto, vários estudos analisam esse tipo de notícia neste cenário.

Notícias Enviesadas: Referem-se às notícias cujo conteúdo é extremamente enviesado ou tendencioso. No contexto político, são conhecidas como notícias hiperpartidárias (POTTHAST et al., 2017) e possuem um conteúdo extremamente tendencioso em relação a uma pessoa/partido/evento.

Sátiras: São histórias que contém a presença de elementos de ironia e humor. Este tipo de notícia vem recebendo considerável atenção na Web nos últimos anos. Alguns exemplos populares de sites que publicam sátiras são, Sensacionalista ³, TheOnion ⁴ e SatireWire ⁵. Geralmente, estes sites evidenciam sua natureza satírica em suas páginas. No entanto, como seus artigos geralmente são disseminados pelas redes sociais, esse fato é ofuscado pelo próprio site ou ignorado pelos usuários que geralmente os levam a sério sem checagem de fonte.

Vale salientar que pode haver sobreposição dos tipos de conteúdos mencionados acima em algumas notícias. Portanto, podem se enquadrar em várias categorias. Um rumor

² <http://www.e-farsas.com/lima-duarte-morre-aos-86-anos-de-idade-verdade-ou-farsa.html>

³ <https://www.sensacionalista.com.br>

⁴ <https://www.theonion.com/>

⁵ <http://www.satirewire.com>

pode, por exemplo, conter uma teoria da conspiração, uma vez que ambas contém fatos não verificados e não provados; sátiras podem conter um texto enviesado, uma vez que a ironia pode conter algum viés contra ou a favor de uma pessoa/partido/evento.

2.1.2 Aspectos Psicológicos

Existem várias teorias psicológicas que podem explicar o fenômeno e o poder das notícias falsas. Normalmente, notícias falsas visam principalmente os consumidores, explorando suas vulnerabilidades individuais. Existem dois fatores principais que tornam os consumidores naturalmente vulneráveis às notícias falsas, que são: (i) *Realismo Ingênuo*: os consumidores tendem a acreditar que suas percepções de realidade são as mais corretas e aqueles que discordam são desinformados, irracionais ou tendenciosos (WARD L ROSS; BROWN, 1997); e (ii) *Confirmação de Viés*: os consumidores preferem receber informações que confirmem seus pontos de vista (NICKERSON, 1998). Por conta destes distúrbios inerentes ao comportamento humano, as *Fake News* são constantemente e facilmente consumidas pela sociedade. Além disso, uma vez que estes comportamentos são incorporados, eles são difíceis de corrigir. Nyhan e Reifler (2010) fazem um estudo que mostra que, especialmente em grupos ideológicos, tentativas de correções de notícias falsas não só falharam como também aumentaram suas crenças nas notícias falsas que comprovam suas ideologias.

Na área de comunicação existe um termo chamado *câmara de eco ideológica*. Consumidores deste grupo são seletivamente expostos a certos tipos de conteúdos em seus *Feeds* de notícias. Estas notícias possuem sempre uma alta afinidade com as ideologias do consumidor. Tufekci (2015) conclui em seu estudo que as pessoas tendem a não clicar em matérias de assuntos nos quais elas discordam: 6% de cliques a menos de indivíduos liberais em matérias conservadoras e 17% de cliques a menos de indivíduos conservadores em matérias liberais.

Além disso, os algoritmos de personalização de conteúdo, presentes em redes sociais, são matematicamente precisos e irão aprender o comportamento que o usuário desejar. Com isso, cabe o seguinte questionamento: *será que as câmaras de eco ideológicas presentes nas redes sociais são de responsabilidade dos algoritmos das redes sociais ou do próprio comportamento humano de se agrupar em tribos ideológicas?* Para Eytan Bakshy, pesquisador do Grupo de Ciência de Dados do Facebook, as escolhas individuais das pessoas acerca do que interagir e do que mais importa para elas, servem de parâmetros para os algoritmos de sugestão de conteúdo. Logo, um usuário que sempre interage com um nicho de conteúdo, com o passar do tempo, induz os algoritmos a apresentarem apenas aqueles conteúdos (BAKSHY; MESSING; ADAMIC, 2015). Ou seja, as câmaras de eco ideológicas são outro local onde a proliferação das notícias falsas são favorecidas pelo fato delas confirmarem as ideologias daqueles que as consomem.

Uma vez compreendido o ecossistema das *Fake News*, bem como os fatores psicológicos de levam pessoas a consumirem estes conteúdos, percebe-se o quão complexo é tratar deste tema pelo lado computacional. Isso corrobora para a importância do objetivo principal deste projeto, que é o de minimizar o impacto negativo da proliferação das notícias falsas na sociedade contemporânea.

2.1.3 Técnicas de Detecção

As estratégias de detecção de notícias falsas propostas pela literatura podem ser divididas entre: manuais, colaborativas (*crowdsourcing*) e automáticas (Figura 1). As estratégias manuais consistem de serviços de *fact-checking* disponibilizados por equipes de comunicação ou empresas especializadas, tais como *Fato ou Fake*⁶, *Snopes*⁷ e *Fact-Check.org*⁸. Isso é uma vantagem e uma desvantagem. Afinal, a confiabilidade desta estratégia está acima de qualquer algoritmo de detecção por se tratar de uma checagem dos fatos feita por uma equipe especializada de jornalistas investigativos. Por outro lado, o fato da checagem ser feita manualmente, torna-a inviável como solução para um mecanismo automatizado por conta de ser uma checagem demorada e manual. Ainda assim, muitas bases de dados utilizadas em experimentos de detecção automática são alimentadas com dados de detecção obtidos por meio desta estratégia.

As estratégias colaborativas consistem de ferramentas ou sistemas de informação com dados alimentados pelos usuários. Ela, notoriamente, é mais rápida que a anterior. Contudo, para que esta estratégia seja viável, é necessário o uso de algoritmos de avaliação de confiabilidade ou de ranqueamento de usuários, afim de minimizar falsas detecções (TACCHINI et al., 2017; KIM et al., 2018; ARIF et al., 2017; TSCHIATSCHEK et al., 2017; RUCHANSKY, 2017; GIUDICE, 2010; SETHI, 2017; JIN et al., 2016).

O escopo deste trabalho compreende apenas estratégias de detecção automática usando algoritmos de Aprendizagem Profunda (*Deep Learning*) com técnicas de NLP para classificação de textos, com o intuito de extrair a veracidade de uma notícia (OSHIKAWA; QIAN; WANG, 2018b). As palavras e sentenças são valoradas de acordo com seu contexto morfosintático, semântico e psicológico. Em seguida, são submetidas à estrutura de Aprendizagem Profunda proposta. A justificativa para esta limitação de escopo se sustenta quando se compara o tempo necessário para uma detecção manual ou colaborativa com uma solução automática. Além disso, detectores automáticos de notícias falsas podem apresentar um valor percentual de veracidade de uma entrada, afastando quaisquer possíveis saídas tendenciosas provenientes de alguns veículos de comunicação ou jornalistas tendenciosos (VARGO; GUO; AMAZEEN, 2018).

⁶ <https://g1.globo.com/fato-ou-fake/>

⁷ <https://www.snopes.com>

⁸ <https://www.factcheck.org>

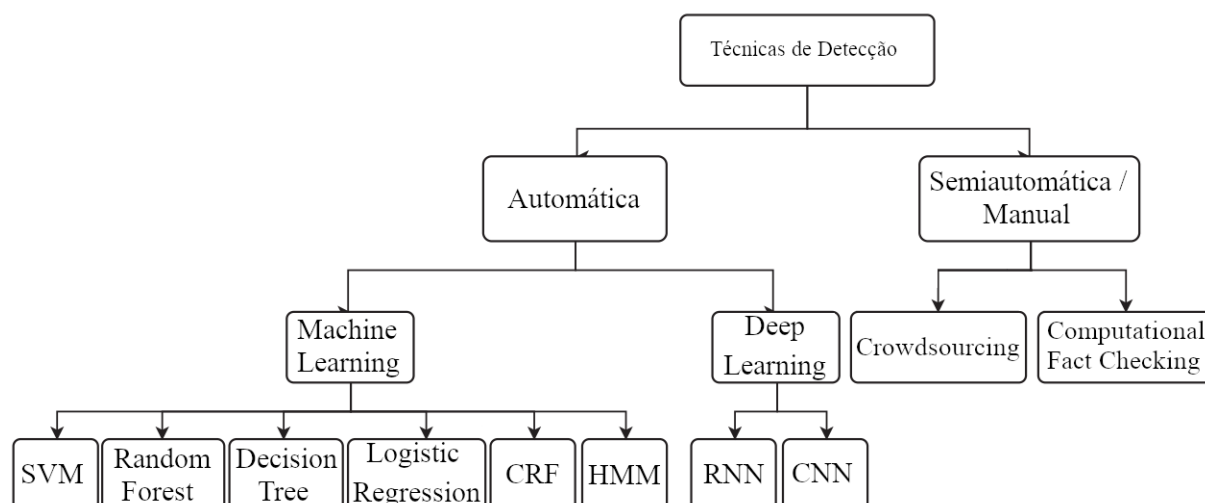


Figura (1) – Diferentes abordagens de detecção de notícias falsas (adaptado de (BONDIELLI; MARCELLONI, 2019a))

2.2 Processamento de Linguagem Natural

O Processamento de Linguagem Natural (do inglês *Natural Language Processing*, ou abreviadamente NLP) é uma subárea da mineração de textos, inteligência artificial e linguística computacional responsável por criar e processar automaticamente a linguagem natural humana (BORDIGNON et al., 2018). Ela considera a estrutura hierárquica da linguagem: várias palavras formam uma frase, várias frases fazem uma sentença e, em última análise, sentenças transmitem ideias (BARBOSA et al., 2017). Uma análise de texto usando NLP, por meio da compreensão e extração de seu significado, pode ser composta por três níveis:

- (i) *Análise Morfológica*, que foca na estrutura das palavras. Neste nível, são utilizadas técnicas, tais como Filtragem, Tokenização, Lematização e Radicalização (ver 2.3);
- (ii) *Análise Sintática*, que trata das relações entre as palavras em uma frase, decidindo qual classe gramatical da palavra no idioma, podendo, até mesmo, inferir por análise de contexto a próxima classe gramatical (INDURKHYA; DAMERAU, 2010). A técnica mais conhecida para este cenário é o uso de Etiquetadores (POS Tagging); e
- (iii) *Análise Semântica*, que é construída com base nos resultados dos outros dois níveis e tem como objetivo definir palavras e sentenças com base no conhecimento de sua estrutura, relacionamento e função. É neste nível onde são usados modelos semânticos de representação distribuída de palavras (ver 2.4).

Sobre aplicações em NLP, pode-se elencar: recuperação de informação (MANNING; RAGHAVAN; SCHÜTZE, 2008), classificação de documentos (JOHNSON; ZHANG, 2015),

similaridade de palavras (PENNINGTON; SOCHER; MANNING, 2014), análise de sentimentos (TANG et al., 2014), sistemas de *Question Answering* (BORDES; CHOPRA; WESTON, 2014), dentre outras.

2.2.1 Técnicas de Pré-Processamento

Para grandes coleções de documentos, é imprescindível realizar um pré-processamento do texto. Isso porque as estruturas de aprendizagem que se propõem a classificar textos trabalham em cima de dados estruturados e não de um texto plano (HOTH; NÜRNBERGER; PAASS, 2005). A maioria das abordagens de mineração de texto se baseiam na ideia de que um documento nada mais é do que um conjunto de palavras. Nesta estrutura, cada palavra é representada por um vetor com sua “importância numérica”. Contudo, até se chegar na vetorização de cada palavra, algumas técnicas de pré-processamento de texto são necessárias.

2.2.1.1 Corpus

Corpus nada mais é do que uma coleção de textos completos ou partes dele contidos em todo o *dataset* de análise com o intuito de estudar e descrever um idioma (MITKOV, 2004). O dicionário, ou vocabulário, é uma lista com todas as palavras presentes no corpus. Através do dicionário é possível criar um vetor, onde o índice representa o valor de cada palavra presente no Corpus. A este vetor dá-se o nome de vetor de características (ou *feature vector*).

2.2.1.2 Filtragem, Tokenização, Lematização e Radicalização

Como o dicionário é apenas um conjunto de palavras inéditas conditas no texto, ele pode ficar grande e, portanto, técnicas são necessárias para a redução do dicionário sem perda significativa de informação útil.

Uma delas é a **Filtragem**. Ela consiste na remoção de stopwords, tais como artigos, conjunções, preposições, etc. Existem também palavras que ocorrem muito raramente e muito frequentemente não tem muita importância no documento. Nesta técnica, métodos de seleção de termos são usadas.

A **Tokenização** visa transformar textos em unidades atômicas e discretas, chamadas *tokens*. Para sistemas de escrita alfabética (tais como Inglês, Português, Francês, Espanhol, dentre outros) este processo não requer complexidade. Entretanto, em sistemas de escrita logográfica, como o Chinês, as palavras são compostas por uma pequena quantidade de símbolos sem estarem separadas por espaços em branco. Nestes casos, é necessário o uso de classificadores de regressão logística (*Logistic Regression*, ou abreviadamente LR) para tornar possível a segmentação das palavras (EISENSTEIN, 2019).

Lematização consiste em encontrar a forma canônica da palavra, através da análise morfológica do texto. Como esse processo pode ser bastante custoso e ainda passível de erros, na prática, os métodos de stemming são aplicados.

Métodos de **Radicalização** (ou “Stemmização”, do inglês *Stemming*) tentam construir a forma básica da palavra, eliminando prefixos e sufixos e encontrando seu radical. Por exemplo, um verbo que aparece várias vezes. Porém, em tempos verbais distintos, não seria armazenado repetidamente no processo de radicalização, reduzindo, assim, o tamanho do vocabulário sem perda de informação.

2.2.1.3 Stopwords

As *stopwords* são palavras muito comuns em qualquer idioma. Na língua portuguesa, por exemplo, artigos definidos (“o”, “a”, “os”, “as”) e preposições (como “de”, “para”) estão presentes em muitas frases. Assim como as palavras flexionadas usadas na Radicalização, a lista de *stopwords* varia pela estrutura morfológica do idioma. Estes termos presentes na lista de *stopwords*, por funcionar predominantemente como um conectivo de palavras, não apresentam relevância no processo de classificação de um documento e podem ser removidas do vocabulário.

Entretanto, existem casos, tais como “não gosto” que quando removido “não” muda o sentido da palavra adjacente. Desta forma, é necessário aplicar filtros à lista de stopwords.

2.3 Parts-of-Speech Tagging

Um etiquetador morfossintático (do inglês *Parts-of-Speech Tagging* ou, abreviadamente, POS-tagging) realiza numa tarefa estruturada onde cada palavra é etiquetada tendo em vista sua participação numa sentença (THANAKI, 2017). As informações geradas através de uma tarefa de etiquetamento podem ser de natureza intrínseca (etiquetamento gramatical) ou extrínseca (remoção de ambiguidades baseada em análise de contexto). A palavra “leve”, por exemplo, pode aparecer numa sentença tanto na forma de adjetivo (“Este livro está muito leve.”), quanto na forma de verbo (“Talvez eu leve o dia todo para terminar este livro”).

Inicialmente é necessário realizar parte (ou todas, caso necessário) das técnicas de pré-processamento descritas anteriormente (ver). Em seguida, deve-se verificar a existência de caracteres “não linguísticos” como URL, caracteres especiais, aspas e travessões. Em suma, no texto “limpo” é feita a tokenização e enviado ao etiquetador, onde cada palavra recebe uma etiqueta conforme sua classe gramatical, limitada por um conjunto de etiquetas disponibilizadas pelo etiquetador.

Em linhas gerais, existem três principais abordagens utilizadas pelos etiquetadores

morfossintáticos: a linguística (baseada em regras), a probabilística (baseada em n-gramas) e híbrida (baseada em regras e em n-gramas).

2.3.1 Brill Tagger

Em seu estudo, Brill (1992) propõe um etiquetador híbrido não supervisionado cujo aprendizado é baseado em regras de transformação dirigidas por erro (do inglês *Transformation-Based Learning*, ou TBL). Embora os métodos predominantemente usados para etiquetagem sejam de natureza estocástica, ele mostra que seu sistema baseado em transformação simples pode funcionar no mesmo nível dos demais etiquetadores.

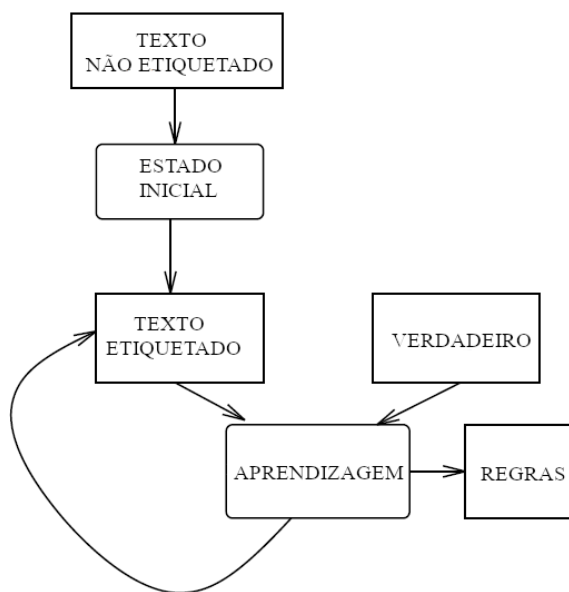


Figura (2) – Etiquetagem segundo o método Brill (BRILL, 1994)

O treinamento de um etiquetador de Brill consiste de duas etapas: aprender regras para etiquetagem de palavras desconhecidas e aprender regras contextuais. Conforme mostrado na Figura 2, o corpus não etiquetado passa, primeiramente, por uma etiquetagem inicial. Após ter sido etiquetado, o corpus é comparado com a etiquetagem correta (um corpus manualmente etiquetado). Então, uma lista de regras de transformação é aprendida afim de ser aplicada novamente ao etiquetador. Cada transformação é composta pela regra descrita e pelo contexto que desencadeia a regra. Uma possível regra para o modelo adaptado ao idioma Português, seria:

Mude a etiqueta da palavra atual de V (verbo) para N (substantivo), se a mesma for precedida da etiqueta ART (artigo).

A cada iteração de aprendizado, a transformação para melhorar a etiquetagem é encontrada de acordo com uma função objetivo utilizada e então adicionada à lista

ordenada de transformações. O aprendizado termina quando não se encontram mais transformações que possam melhorar o corpus.

Por sua característica não supervisionada e orientada a regras de transformação, muitos etiquetadores voltados para o Português foram propostos usando como base o método de Brill (SANTOS; MILIDIÚ; RENTERÍA, 2008; KINOSHITA; SALVADOR; MENEZES, 2006; KINOSHITA et al., 2007; FINGER, 2000; MENEZES; NETO, 2002). Este trabalho utiliza o modelo de Brill treinado no corpus MAC-Morpho (ver 2.3.1.1)

2.3.1.1 Corpus MAC-Morpho

O MAC-Morpho é um corpus com textos extraídos do jornal Folha de São Paulo⁹ desde 1994, nos mais diversos cadernos do jornal (dentre eles, Esportes, Ciência, Economia, Informática, Mundo, Política, etc). Sua primeira versão foi em 2003 (ALUÍSIO et al., 2003), seguida de duas revisões, sendo estas em 2013 e 2015 (FONSECA; ROSA, 2013; FONSECA; ROSA; ALUÍSIO, 2015). Atualmente o MAC-Morpho disponibiliza um *tagset* que abrange satisfatoriamente a escrita em Português, conforme mostrado na Tabela 1, e conta com um banco de mais de 1,1 milhão de palavras etiquetadas manualmente pelo Projeto Lacio-Web¹⁰, do Grupo NILC-USP (Núcleo Interinstitucional de Linguística Computacional da USP). Este corpus é utilizado como referência de base de treinamento para propostas de etiquetadores morfossintáticos em Português.

2.3.2 NLPNet

Produzido pelo grupo NILC-USP, este etiquetador se baseia em Redes Neurais Artificiais para realizar a tarefa de etiquetagem¹¹. Foi inspirado na ferramenta SENNA¹² que baseia seu etiquetador no modelo de Collobert et al. (2011). Este etiquetador possui uma abordagem baseada em Anotação de Regras Semânticas (do inglês *Semantic Role Labeling*, ou abreviadamente SRL), que atesta que: dadas uma sentença e um verbo a ela designado, deve-se identificar os argumentos do verbo e anotá-los com uma etiqueta que defina a relação semântica entre o verbo e seus argumentos (MÀRQUEZ et al., 2008). Neste modelo, as relações semânticas são preenchidas de acordo com as especificidades do idioma através de uma lista de regras pré-estabelecidas. Um exemplo de regra para o idioma Português seria na sentença “*João sorriu para Maria*”. Esta sentença poderia ser anotada com os papéis semânticos *agente* e *destinatário*:

[*João*]_{agente} [*sorriu*]_{verbo} para [*Maria*]_{destinatário}

⁹ <https://www.folha.uol.com.br/>

¹⁰ www.nilc.icmc.usp.br/nilc/projects/lacio-web.htm

¹¹ <http://nilc.icmc.usp.br/nlpnet/>

¹² <http://ronan.collobert.com/senna/>

CLASSE GRAMATICAL	ETIQUETA
ADJETIVO	ADJ
ADVÉRBO	ADV
ADVÉRBO CONECTIVO SUBORDINATIVO	ADV-KS
ADVÉRBO RELATIVO SUBORDINATIVO	ADV-KS-REL
ARTIGO	ART
CONJUNÇÃO COORDENATIVA	KC
CONJUNÇÃO SUBORDINATIVA	KS
INTERJEIÇÃO	IN
NOME	N
NOME PRÓPRIO	NPROP
NUMERAL	NUM
PARTICÍPIO	PCP
PALAVRA DENOTATIVA	PDEN
PREPOSIÇÃO	PREP
PRONOME ADJETIVO	PROADJ
PRONOME CONECTIVO SUBORDINATIVO	PRO-KS
PRONOME PESSOAL	PROPESS
PRONOME RELATIVO CONECTIVO SUBORDINATIVO	PRO-KS-REL
PRONOME SUBSTANTIVO	PROSUB
VERBO	V
VERBO AUXILIAR	VAUX
SÍMBOLO DE MOEDA CORRENTE	CUR
ETIQUETAS COMPLEMENTARES (Estrangeirismos; Apostos; Dados; Números de Telefone; Datas; Horas; e Disjunção)	 EST AP DAD TEL DAT HOR [[]]
CONTRAÇÕES e ÊNCLISES	 +
MESÓCLISES	!

Tabela (1) – *Tagset* do corpus MAC-Morpho

Este trabalho utiliza o NLPNet disponibilizado pelo grupo NILC-USP, cujo etiquetador fora treinado no corpus MAC-Morpho (ver 2.3.1.1).

2.3.3 LX-Tagger

LX-Tagger é um etiquetador estatístico e morfossintático criado pelo LX-Center ¹³, vinculado ao NLX (Grupo de Fala e Linguagem Natural, do Departamento de Informática da Universidade de Lisboa). Sua construção tem por base o modelo de etiquetagem estatística de Máxima Entropia, proposto por Ratnaparkhi (1996). Branco e Silva (2004) mostram que o LX-Tagger alcança uma precisão de 97,08% ao utilizar este modelo. Além disso, neste cenário, o autor reporta que foram encontrados 2% de *tokens* ambíguos, dentre os quais foram resolvidos 99,04% dos casos.

2.3.3.1 LX-Corpus

O corpus utilizado no LX-Tagger é o LX-Corpus, que trata-se de um subconjunto do corpus CINTIL ¹⁴. O LX-Corpus conta com, aproximadamente, 230 mil palavras presentes nas mais de 12 mil sentenças advindas de notícias e diálogos escritos em português (BRANCO; SILVA, 2004). Ele gerou um *tagset* para o LX-Tagger que pode ser visto na Tabela 2

Além do LX-Tagger, o LX-Center também possui outras contribuições para o Processamento de Linguagem Natural no idioma Português, tais como o LX-Tokenizer (tokenizador), LX-Gram (uma gramática computacional do Português) e o LX-Lemmatizer (lemmatizador automático para verbos em Português). Todas estas ferramentas estão disponíveis online através de seu sítio eletrônico.

2.3.4 HunPos

HunPos é uma implementação em linguagem OCaml¹⁵ do etiquetador estatístico Trigrams'n'Tags, ou TnT (BRANTS, 2000). Ele apresenta resultados competitivos tanto em precisão quanto em velocidade de etiquetamento. Assim como o TnT, ele também se baseia no Modelo Oculto de Markov (do inglês *Hidden Markov Model*, ou abreviadamente HMM) de segunda ordem, que estima as probabilidades de um dado trigramma para uma dada sequência de palavras. Contudo, diferente dos demais etiquetadores HMM, o HunPos calcula as probabilidades usando a *tag* atual quanto as *tags* anteriores, o que torna a taxa de erro 10% menor que os demais etiquetadores HMM (HALÁCSY; KORNAI; ORAVECZ, 2007). Além disso, em comparação com os etiquetadores descritos neste trabalho, o HunPos possui a vantagem de ser treinado com diferentes *tagsets* e em diferentes *corporas*.

Para palavras desconhecidas, o HunPos gera todos os rótulos possíveis e, em seguida, atribui pesos a eles por meio de um algoritmo de adivinhação de sufixos, com base na distribuição de termos de baixa ocorrência no corpus. Banko e Moore (2004) comprovam

¹³ <http://lxcenter.di.fc.ul.pt/>

¹⁴ <http://cintil.ul.pt/cintilfeatures.html>

¹⁵ <https://ocaml.org/learn/>

CLASSE GRAMATICAL	ETIQUETA	CLASSE GRAMATICAL	ETIQUETA
Adjetivo	ADJ	Classe de Magnitude	MGT
Advérbio	ADV	Meses	MTH
Cardinal	CARD	Sintagma Nominal	NP
Conjunção	CJ	Ordinal	ORD
Clíticos	CL	Parte de Endereço	PADR
Nome	CN	Parte de Nome Próprio	PNM
Artigo Definido	DA	Pontuação	PNT
Demonstrativo	DEM	Possessivos	POSS
Fracções	DFR	Particípio Passado (exceto termos compostos)	PPA
Números Romanos	DGTR	Sintagma Preposicional	PP
Dígitos	DGT	Particípio Passado (termos compostos)	PPT
Marcadores de Discurso	DM	Preposição	PREP
Endereço Eletrónico	EADR	Pronome Pessoal	PRS
Final de Enumeração	EOE	Quantificadores	QNT
Exclamativa	EXC	Pronome Relativo	REL
Gerúndio	GER	Títulos Sociais	STT
Gerúndio	GERAUX	Símbolos	SYB
Artigo Indefinido	IA	Terminações opcionais	TERM
Indefinidos	IND	"um"ou "uma"	UM
Infinitivo	INF	Abrev. de unidade de medida	UNIT
Infinitivo	INFAUX	Verbos "ter"ou "haver" em forma verbal composta	VAUX
Interrogativos	INT	Verbos	V
Interjeição	ITJ	Dias da Semana	WD
Letras	LTR		
EXPRESSÕES MULTI-PALAVRAS (LOCUÇÕES)			
CLASSE GRAMATICAL	ETIQUETA		
Locuções Advérbiais	LADV1...LADV_n		
Locuções Conjuntivas	LCJ1...LCJ_n		
Locuções Demonstrativos	LDEM1...LDEM_n		
Locuções Numéricas (Fração)	LDFR1...LDFR_n		
Locuções Discursivas	LDM1...LDM_n		
Locuções Interjeitivas	LITJ1...LITJ_n		
Locuções Pronominais Pessoais	LPRS1...LPRS_n		
Locuções Prepositivas	LPREP1...LPREP_n		
Locuções Numéricas	LQD1...LQD_n		
Locuções Relativos	LREL1...LREL_n		

Tabela (2) – Tagset do corpus LX-Corpus

que o tamanho do léxico morfológico fornecido ao etiquetador HunPos impacta diretamente na sua performance para palavras desconhecidas.

2.4 Modelos de Representação Textual

Conforme dito anteriormente, os algoritmos computacionais entendem apenas dado estruturado. Logo, uma tarefa importante em NLP é o modelo representação textual. Existem duas formas de representação textual: representação de documentos e representação distribuída de palavras.

2.4.1 Representação Vetorial de Documentos

Esta é a forma mais comum de representação textual, conhecida como modelo espaço-vetorial (*Vector Space Model* - VSM). Neste modelo, cada palavra w pode ter um peso associado à sua significância dentro do documento. Um documento d pode, então, ser visto como vetor em um espaço n -dimensional, onde n é o número de palavras diferentes existentes no documento dentro de uma coleção de documentos D que define todo o corpus. Ou seja, na prática, o VSM cria representações vetoriais a partir de documentos textuais. A coleção de palavras neste espaço vetorial é chamada de sacola de palavras (*Bag of Words* - *BoW*).

Pode-se notar que, nesta representação, o documento é representado por um vetor formado pelo peso de cada palavra. Logo, nota-se uma característica que, dependendo da aplicação, pode ser considerada uma limitação: a ordem das palavras no documento é desconsiderada.

A Figura 3 representa graficamente o VSM. Nota-se que cada eixo $t_1, t_2, \dots, t_n$ é um termo dentro do VSM. Já o vetor de documento, (d_n) , é composto pelas componentes destes termos. Estas componentes podem ser criadas a partir da presença ou ausência de termos (*One-Hot Encoding*), ou a partir da frequência absoluta ou relativa desses termos (*Term Frequency-Inverse Document Frequency* - TF-IDF).

Neste contexto de espaço vetorial, pode-se imaginar que vetores próximos são interpretados como documentos similares. A Figura 3 retrata este cenário, onde uma consulta é representada através de um vetor q no espaço VSM e o retorno desta consulta são os vetores mais próximos de q no espaço vetorial. Este princípio é a base para uma das principais tarefas de NLP: a Recuperação de Informação (*Information Retrieval* - IR). Sistemas de mecanismo de busca se baseiam nesta medida de similaridade de vetores de documentos em relação a um vetor de consulta. Desta forma, quando um usuário realiza uma consulta, o motor de busca retorna uma lista de vetores de documentos ordenada por similaridade no espaço VSM.

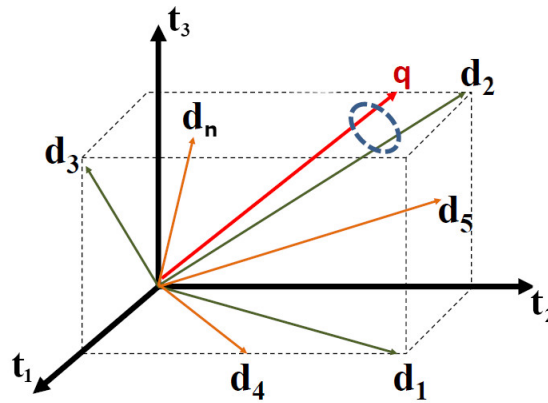


Figura (3) – Representação gráfica do modelo VSM (adaptado de (ZHAI; MASSUNG, 2016))

2.4.1.1 One-Hot Encoding

A representação VSM mais clássica é a que define um vetor, chamado de *one-hot*. Ou seja, este vetor é do tamanho do vocabulário, baseado em representação binária, que atribui peso 1 quando a palavra está presente no documento e 0, caso contrário.

É possível, então, gerar uma matriz de co-ocorrência esparsa, onde as linhas são os documentos e as colunas são as palavras.

2.4.1.2 Term Frequency-Inverse Document Frequency (TF-IDF)

O TF-IDF é uma versão ponderada da *One-Hot Encoding* cuja premissa se baseia no cálculo de frequência de um termo (*term frequency* - *tf*) e da frequência inversa do documento (*inverse document frequenc* - *idf*). A representação da frequência de um termo t em um documento d é dada por $tf(t, d)$. Já o $idf(t, D)$ é dado por

$$idf(t, D) = \log \frac{|D|}{|d \in D : t \in d|}, \quad (2.1)$$

onde $|D|$ é a quantidade de documentos e o denominador da fração presente no logaritmando representa o número de documentos em que o termo t aparece. Logo a matriz de co-ocorrência gerada pelo TF-IDF é dada por

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \quad (2.2)$$

2.4.2 Representação Distribuída das Palavras

Outra forma de representação textual se baseia na ideia de distribuir semanticamente as palavras em um espaço vetorial. Essa representação mapeia o significado das palavras

em um espaço vetorial de baixa dimensão, chamado de *embedding space* (GOLDBERG, 2017).

A geração destes vetores de palavras se dá de duas formas: baseada em contagem, como em *Latent Semantic Analysis* (LSA) (DEERWESTER et al., 1990); e baseada em métodos preditivos, como em Word2Vec (MIKOLOV et al., 2013) (e suas variações CBoW e Skip-Gram).

Ao contrário da representação VSM mostrada anteriormente, a representação em palavras se baseia na hipótese da distribuição, abordada no trabalho de Harris (1954). Nela, o autor afirma que *palavras são similares se aparecem em contextos similares*. Ou seja, este processo cria e refina a matriz de co-ocorrência a partir dos próprios atributos de treinamento, onde as linhas representam uma palavra dentro do corpus e as colunas são os possíveis contextos em que as palavras foram observadas dentro de uma janela em torno da palavra analisada. As famílias de algoritmos de Collobert e Weston (2008), Collobert et al. (2011), Word2Vec e GloVe (descritas a seguir) foram inspiradas nesta ideia.

2.4.2.1 Latent Semantic Analysis (LSA)

Um problema conhecido na utilização da hipótese BoW está na hiper-dimensionalidade das matrizes de co-ocorrência. Considere um corpus para um problema real que seja composto de, pelo menos, algumas centenas de documentos. Isso fará com que o número de termos do vocabulário ultrapasse as centenas de milhares. Logo, a matriz de co-ocorrência possuirá um tamanho impraticável computacionalmente. Este problema comum em NLP é chamado de maldição da dimensionalidade (*curse of dimensionality*).

Para contornar este problema de dimensão da matriz de co-ocorrência, recorre-se a um método conhecido como *Latent Semantic Analysis* (LSA). Após a criação da matriz de co-ocorrência, é feita uma redução da sua dimensionalidade através de um método de decomposição, chamado *Singular Value Decomposition* (SVD) (DUMAIS et al., 1988).

2.4.2.2 Singular Value Decomposition (SVD)

A Decomposição em Valores Singulares (ou *Singular Value Decomposition* - SVD) é uma forma muito importante de decomposição de matrizes, tendo diversas aplicações (dentre elas a redução dimensional aplicada à problemas de NLP). Segue abaixo seu teorema:

Theorem 2.4.1 *Se A é uma matriz real de dimensão $m \times n$ ($A \in \mathbb{R}^{m \times n}$), existem matrizes ortogonais*

$$U = [u_1 \dots u_m] \in \mathbb{R}^{m \times m} \text{ e } V = [v_1 \dots v_n] \in \mathbb{R}^{n \times n}$$

tais que

$$U^T A V = \Sigma = \text{diag}(\sigma_1, \dots, \sigma_p) \in \mathbb{R}^{m \times n}, p = \min \{m, n\},$$

onde $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_p \geq 0$.

Em linhas gerais, esta redução preserva as principais características do modelo anterior fazendo uma generalização. Contudo, invariavelmente, algumas características da matriz original serão perdidas em troca de um ganho de redução dimensional.

2.4.3 Métodos Preditivos

Nesta categoria estão os métodos que tentam prever uma palavra a partir de seus vizinhos, utilizando algoritmos de aprendizagem de máquina. Os vetores gerados são chamados de Vetores de Palavras Embutidas (*Word Embeddings*) e são considerados como parâmetros do modelo. É importante destacar que, apesar de utilizar técnicas de aprendizagem supervisionada, o processo de treinamento para geração das *word embeddings* é considerada uma tarefa não-supervisionada (GOLDBERG, 2017).

Destacam-se como exemplos de métodos preditivos, o Modelo de Linguagem de Redes Neurais (*Neural Network Language Model – NNLM*) (BENGIO et al., 2003), o método *Global Vectors for Word Representation (GloVe)* (PENNINGTON; SOCHER; MANNING, 2014) e o Word2Vec (nas versões *CBoW* e *Skip-Gram*)(MIKOLOV et al., 2013; MIKOLOV et al., 2013; MIKOLOV; LE; SUTSKEVER, 2013).

2.4.3.1 Word2Vec

Este é um dos modelos preditivos mais utilizados em problemas de NLP. Ele foi desenvolvido por Tomas Mikolov e seus colegas em uma série de artigos (MIKOLOV et al., 2013; MIKOLOV et al., 2013; MIKOLOV; LE; SUTSKEVER, 2013). Assim como no algoritmo proposto por Collobert e Weston (2008), Word2Vec também inicia com uma arquitetura em redes neurais e vai modificando-a até produzir resultados mais rápidos. Ou seja, ao invés de contar palavras e gerar uma matriz de co-ocorrência, este modelo vai refinando o vetor de palavras através de redes neurais, capturando indiretamente as características linguísticas de uma linguagem natural. Em outras palavras, Word2Vec não é um único algoritmo: é um conjunto de implementações que conta com duas representações diferentes (*CBoW* e *Skip-Gram*, que são detalhadas a seguir) e dois métodos de otimização (Amostragem Negativa e SoftMax Hierárquico).

Embora não faça uso de modelos de aprendizagem profunda e sim RNAs de arquitetura rasa, o Word2Vec é utilizado amplamente em problemas que requerem o uso de arquiteturas de *Deep Learning*. A motivação para seu uso nestes tipos de problemas está no fato de a saída do Word2Vec ser facilmente entendida e utilizada por estes modelos.

Geometricamente, o Word2Vec agrupa palavras semanticamente semelhantes no espaço vetorial. Ou seja, o algoritmo consegue criar uma representação distribuída das

palavras de forma eficiente, preservando as relações semânticas que elas possuem com as palavras vizinhas e gerando, assim, *clusters* semânticos (ver Figura 4).

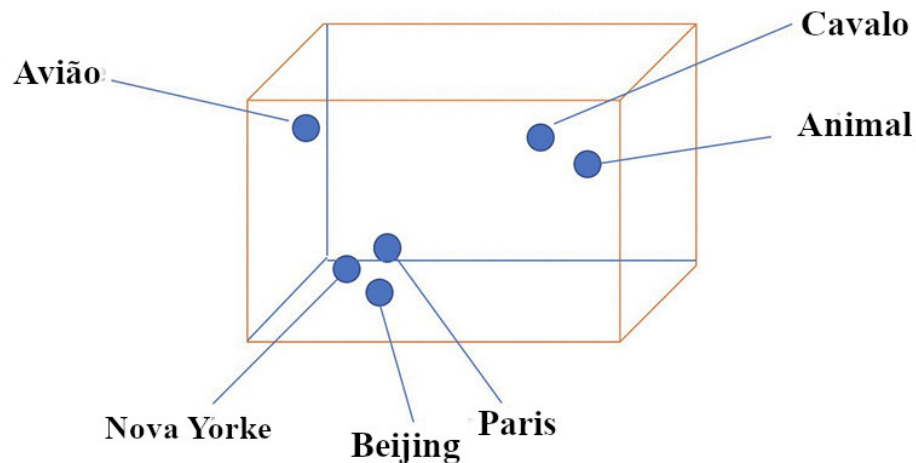


Figura (4) – Word2Vec: palavras que são semanticamente semelhantes estão próximas no espaço vetorial (Fonte: (AMAZON, 2020))

Outro resultado importante obtido a partir do trabalho de Mikolov et al. (2013), Mikolov et al. (2013), Mikolov, Le e Sutskever (2013) reside na ideia de que é possível identificar vetores semânticos, realizando operações entre palavras, como na operação abaixo (mostrada geometricamente na Figura 5):

$$v(rei) - v(rainha) = v(homem) - v(mulher) \quad (2.3)$$

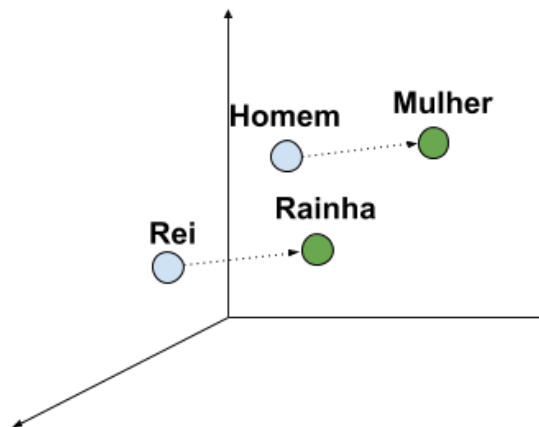


Figura (5) – Similaridade entre vetores que se relacionam em um contexto semântico, chamado gênero (vetor pontilhado)

2.4.3.1.1 Continuous Bag-of-Words (CBoW)

O *Continuous Bag-of-Words* (CBoW) é uma rede neural que prediz uma palavra dado um contexto de palavras vizinhas compreendidas em uma janela. Nesta variante do Word2Vec, a camada de entrada é um vetor *one-hot* com as palavras vizinhas; na camada escondida está o funcionamento do algoritmo; por fim, a camada de saída representa a palavra predita.

Neste cenário, V é o tamanho do vocabulário e a camada escondida possui tamanho N . Assim, a matriz de pesos entre a camada de entrada e a de saída pode ser representada por uma matriz $\mathbf{W}$ $V \times N$. Logo, a camada escondida pode ser dada pelo produto entre a camada de entrada e a média dos vetores de contexto na matriz de pesos

$$\begin{aligned} \mathbf{h} &= \frac{1}{C} \mathbf{W}^T (\mathbf{x}_1 + \mathbf{x}_2 + \dots + \mathbf{x}_C) \\ &= \frac{1}{C} (\mathbf{v}_{w_1} + \mathbf{v}_{w_2} + \dots + \mathbf{v}_{w_C})^T \end{aligned} \quad (2.4)$$

onde C é o número de palavras no contexto, $w_1, w_2, \dots, w_C$ são as palavras presentes neste contexto e $\mathbf{v}_{w_i}$ é o vetor de contexto da palavra w_i .

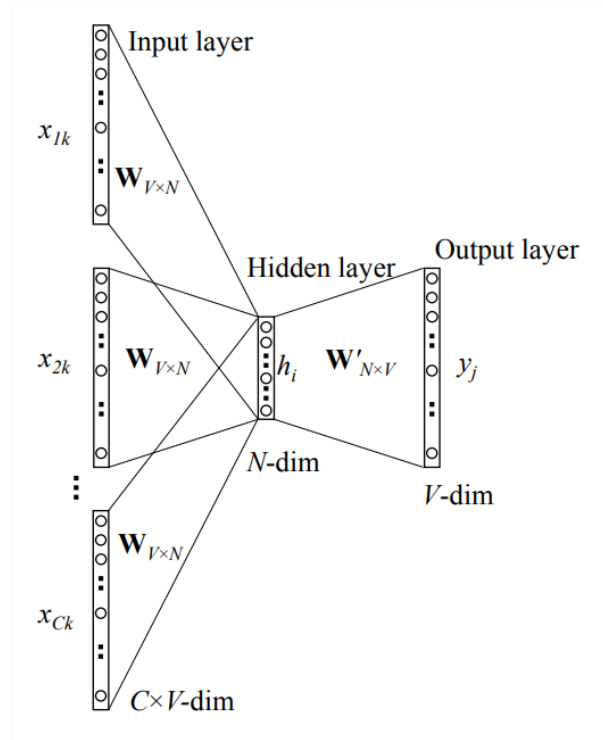


Figura (6) – Modelo CBoW (RONG, 2014)

A Figura 6 ilustra o funcionamento do modelo CBoW. É possível notar que, da camada escondida até a saída, existe uma nova matriz de pesos $\mathbf{W}' = w'_{ij}$ $N \times V$. A

partir destes pesos, podemos calcular a pontuação u_j para cada palavra no vocabulário,

$$u_j = \mathbf{v}_{w_j}'^T \mathbf{h}, \quad (2.5)$$

onde $\mathbf{v}_{w_j}$ é a j -ésima coluna da matriz $\mathbf{W}'$. Usando a otimização SoftMax, obtém-se a seguinte distribuição das palavras:

$$p(w_j|w_I) = \frac{\exp(\mathbf{v}_{w_j}'^T \mathbf{v}_{w_I})}{\sum_{j'=1}^V \exp(\mathbf{v}_{w_j}'^T \mathbf{v}_{w_I})} \quad (2.6)$$

Derivando a matriz de pesos entre a camada escondida e a saída, simplificações matemáticas são obtidas na análise do modelo sem perda de generalização. Assim, a função de perda é dada por

$$\begin{aligned} E &= -\log p(w_j|w_{I,1}, \dots, w_{I,C}) \\ &= -u_{j*} + \log \sum_{j'=1}^V \exp(u_{j'}) \\ &= -\mathbf{v}_{w_O}'^T \cdot \mathbf{h} + \log \sum_{j'=1}^V \exp(\mathbf{v}_{w_j}'^T \cdot \mathbf{h}) \end{aligned} \quad (2.7)$$

Usando Gradiente Descendente Estocástico (para maiores detalhes, ver o Anexo A.1) em cada palavra $w_{I,c}$ presente no contexto dado na entrada I , temos que a atualização da matriz de pesos entre camada escondida $\rightarrow$ saída é dada por

$$\mathbf{v}_{w_j}'^{(novo)} = \mathbf{v}_{w_j}'^{(velho)} - \eta \cdot e_j \cdot \mathbf{h} \quad \text{para } j = 1, 2, \dots, V \quad (2.8)$$

Para a atualização da matriz de pesos entre entrada $\rightarrow$ camada escondida, temos

$$\mathbf{v}_{w_{I,c}}^{(novo)} = \mathbf{v}_{w_{I,c}}^{(velho)} - \frac{1}{C} \cdot \eta \cdot EH^T \quad \text{para } c = 1, 2, \dots, C \quad (2.9)$$

onde $\mathbf{v}_{w_{I,c}}$ é a entrada da c -ésima palavra no contexto dado na entrada e η é uma taxa de aprendizagem positiva. EH^T é a soma dos vetores de saída de todos os vetores do vocabulário V ponderada pela predição do erro, ou seja

$$EH_i = \sum_{j=1}^V e_j \cdot w'_{ij} \quad (2.10)$$

As Equações 2.8 e 2.9 são a base de funcionamento do algoritmo CBoW no caso geral, onde, dado um contexto de entrada com C palavras, espera-se encontrar a palavra correta.

2.4.3.1.2 Skip-Gram

O modelo *Skip-Gram* possui um algoritmo cujo funcionamento é exatamente o inverso do CBoW. O *Skip-Gram* visa prever o contexto que rodeia uma dada palavra. Ou seja, neste modelo a palavra é a camada de entrada e o contexto é a camada de saída (conforme ilustrado na Figura 7).

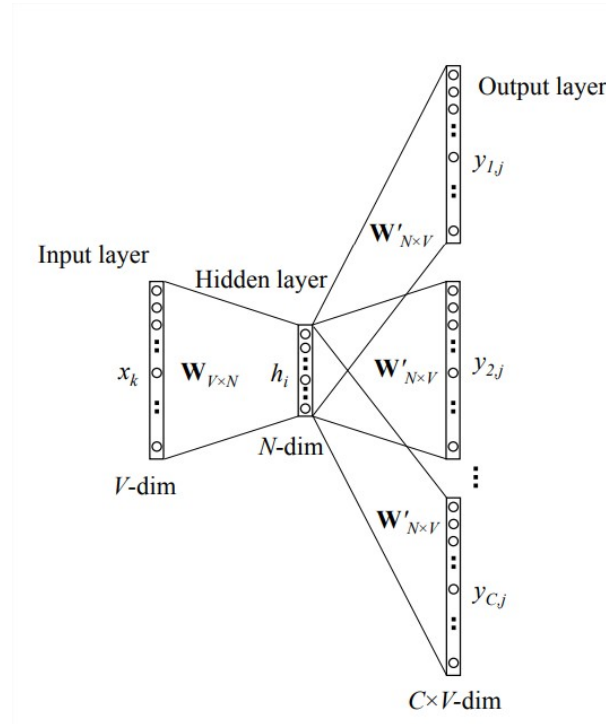


Figura (7) – Modelo Skip-Gram (RONG, 2014)

Essa analogia resulta em definições matematicamente similares para os cálculos de atualizações das matrizes de peso. Na atualização da matriz de pesos entre camada escondida $\rightarrow$ saída, tem-se que

$$\mathbf{v}_{w_j}^{(novo)} = \mathbf{v}_{w_j}^{(velho)} - \eta \cdot EI_j \quad \text{para } j = 1, 2, \dots, V \quad (2.11)$$

onde EI_j é a soma das predições de erro de todas as palavras no contexto a ser encontrado, ou seja

$$EI_j = \sum_{c=1}^C e_{c,j} \quad (2.12)$$

A 2.9 que atualiza a matriz de pesos entre entrada $\rightarrow$ camada escondida, pode ser copiada para o modelo Skip-Gram considerando $C = 1$ (ou seja, considerando que 1 palavra foi dada, que é o cenário do Skip-Gram). Logo, a matriz de pesos é simplificada para a seguinte expressão:

$$\mathbf{v}_{w_I}^{(novo)} = \mathbf{v}_{w_I}^{(velho)} - \eta \cdot EH^T \quad (2.13)$$

2.4.3.1.3 CBoW vs. Skip-Gram

Apesar dos dois algoritmos serem bastante similares, Mikolov et al. (2013) sugerem que o Skip-Gram funciona melhor em tarefas de análise semântica, enquanto que o CBoW apresenta melhores resultados em tarefas de análise sintática. Esta constatação dos autores dá um direcionamento inicial do uso do Skip-Gram na geração da matriz de pesos deste trabalho. Contudo, como o estudo de detecção de notícias falsas em português ainda é incipiente (esta informação será detalhada no Capítulo 3), o uso do CBoW não será desconsiderado.

Além disso, vários trabalhos sugerem que o Skip-Gram se comporta melhor para grandes *datasets* em relação ao CBoW (GANEGEDARA, 2018; MIKOLOV et al., 2013; PENNINGTON; SOCHER; MANNING, 2014). Esta afirmação apresenta credibilidade se considerar que os trabalhos de Mikolov et al. (2013) e Pennington, Socher e Manning (2014) possuem um *corpora* de bilhões de palavras; além de serem os próprios autores dos algoritmos Word2Vec e GloVe, respectivamente.

2.4.4 GloVe

O GloVe (*Global Vectors for Word Representation*) é um modelo global de regressão que combina as vantagens da Fatoração de Matrizes e da Janela de Contexto Local (que são a base dos algoritmos CBoW e Skip-Gram) (PENNINGTON; SOCHER; MANNING, 2014). Os autores provam que a razão entre as probabilidades de co-ocorrência apresenta certos aspectos semânticos.

Seja $P_{ij} = P(j|i)$ a probabilidade de uma palavra j aparecer no contexto da palavra i . Sejam w_i e w_j palavras num contexto e $\tilde{w}_k$ uma palavra em outro contexto. Tem-se que a função de associação é dada por

$$F(w_i, w_j, \tilde{w}_k) = \frac{P_{ik}}{P_{jk}}, \quad (2.14)$$

Através de um conjunto de deduções similares às que foram detalhadas em 2.4.3.1, Pennington, Socher e Manning (2014) chegaram à seguinte função de custo para o GloVe:

$$J = \sum_{i,j=1}^V f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2, \quad (2.15)$$

onde V é o tamanho do vocabulário, b são os vieses e $f(X_{ij})$ é uma função de ponderação na função de custo. A função de ponderação é dada por

$$f(x) = \begin{cases} (x/x_{max})^\alpha & \text{se } x < x_{max} \\ 1 & \text{caso contrário.} \end{cases} \quad (2.16)$$

Nos experimentos, Pennington, Socher e Manning (2014) mostram que o modelo GloVe obteve uma melhor performance sobre os modelos comparados CBoW, SVD e Skip-Gram. Para outros cenários de teste, o GloVe demonstrou menor tempo de treinamento em relação ao Word2Vec, dependendo da configuração do corpus, vocabulário e dimensão do vetor. Estes resultados mostram que o modelo GloVe é um modelo a ser considerado em comparação com Word2Vec nos experimentos deste trabalho.

2.5 Linguistic Inquiry and Word Count (LIWC)

Conforme discutido anteriormente, o conjunto de palavras de um idioma é conhecido por dicionário ou léxico. Elas carregam informações morfológicas, semântica e, algumas vezes, até sintáticas. Todavia, em uma tarefa de detecção de notícias falsas ou tendenciosas, apenas a informação semântica não é suficiente, é necessário identificar discursos de ódio, por exemplo, ou encontrar palavras com valor sentimental elevado e atribuir um peso maior a elas. Para estas tarefas, de identificação de emoções em textos, o conjunto de palavras que carregam informações de emoções é chamado de dicionário afetivo de palavras (FREITAS, 2013).

O LIWC¹⁶ foi criado no trabalho de Pennebaker, Francis e Booth (2001) com o intuito de facilitar a geração de dicionários afetivos. Trata-se de um programa que faz a leitura de textos e busca palavras que são analisadas e rotuladas conforme o dicionário afetivo configurado para a tarefa. No dicionário do LIWC cada palavra possui todos os possíveis rótulos de emoção, que foram preenchidos e julgados por avaliadores linguísticos. Quando o LIWC foi criado, o objetivo era fazer com que uma máquina, ao ler um texto, fosse capaz de calcular o percentual de palavras com emoções positivas e negativas. Com os resultados promissores, o autor passou a incluir novas categorias afetivas (TAUSCZIK; PENNEBAKER, 2010), tais como raiva, tristeza, afeto, dentre outras.

¹⁶ <http://liwc.wpengine.com/>

Filho, Pardo e Aluísio (2013) desenvolveram um estudo analisando a versão brasileira do LIWC para análise de sentimentos criada pelo Núcleo Interinstitucional de Linguística Computacional ¹⁷ (NILC-USP). O *Brazilian Portuguese LIWC* (FILHO; PARDO; ALUÍSIO, 2013) foi construído inicialmente na versão 2001 do LIWC e foi atualizado para as versões 2007 e 2015 do programa. Foi feita através da tradução do dicionário original do programa por três equipes de tradução que utilizaram vários dicionários bilíngues português-inglês. As conjugações foram inseridas automaticamente por meio do dicionário Unitex-PB¹⁸ do mesmo grupo NILC-USP. A versão brasileira do LIWC na sua última versão, possui um léxico de 127.600 entradas, categorizadas em 64 classes, tais como saúde (7.003 entradas), dinheiro (5.353 entradas), corpo (4.766 entradas), amizade (679 entradas), família (96), dentre outras. Este trabalho utiliza todas as 64 classes da última versão brasileira do LIWC como uma das entradas para o modelo de detecção.

2.6 Redes Neurais Artificiais

Uma Rede Neural Artificial (RNA) é um modelo probabilístico cuja ideia para sua criação foi baseada no funcionamento dos neurônios que transmitem informações quando certo tipo de sinal for atendido, provocando o repasse da informação. Ela possui sua utilização em diversas aplicações de aprendizagem de máquina e sua estrutura é muito usado para processar grandes quantidades de dados de treinamento a fim de classificá-los ou identificá-los com alta precisão (BASHEER; HAJMEER, 2000). Sua arquitetura geral é composta por camadas de neurônios artificiais de processamento e conexões entre esses neurônios, conforme apresentado na Figura 8.

A camada de entrada é responsável por receber os padrões, estes são repassadas para que a camada oculta possa capturar/extrair a informação. A camada de saída exibe a distribuição da probabilidade do padrão, podendo ser interpretado em um problema de classificação como sendo a classe pertencente àquele padrão.

2.6.1 Neurônios Artificiais

Para entender as arquiteturas de Redes Neurais Artificiais, é preciso entender o funcionamento dos neurônios artificiais. Como o próprio termo explica, o neurônio artificial é um modelo matemático que visa simular o mecanismo do cérebro humano, mais especificamente a sinapse animal (GOLDBERG, 2017).

Com base neste conceito, pode-se adotar a estrutura matemática apresentada na Figura 9. Considerando que um neurônio artificial é uma unidade computacional com entradas e saídas, atribui-se a cada entrada um peso. Em seguida, o neurônio multiplica a

¹⁷ <http://www.nilc.icmc.usp.br/>

¹⁸ <http://www.nilc.icmc.usp.br/nilc/projects/unitex-pb/web/dicionarios.html>

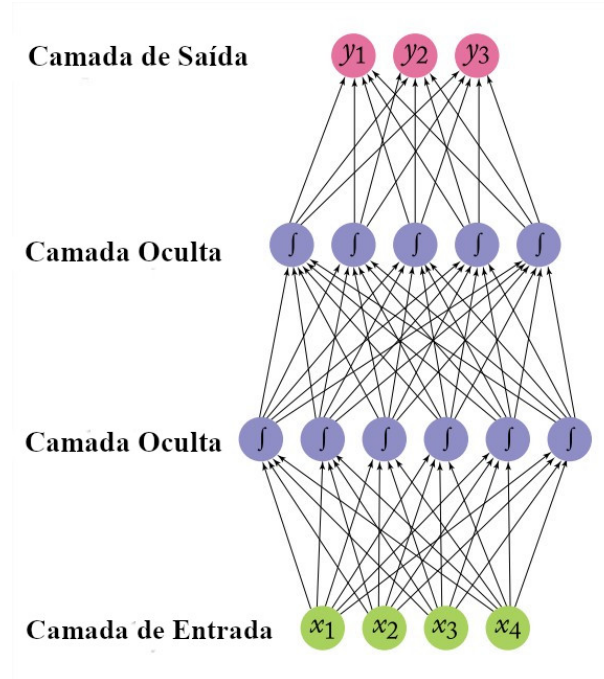


Figura (8) – RNA com duas camadas escondidas (Fonte: Deng e Liu (2018))

entrada ao peso atribuído e adiciona um parâmetro que desloca a função de saída, chamado *bias*. Finalmente, esse resultado é aplicado a um bloco chamado, função de ativação.

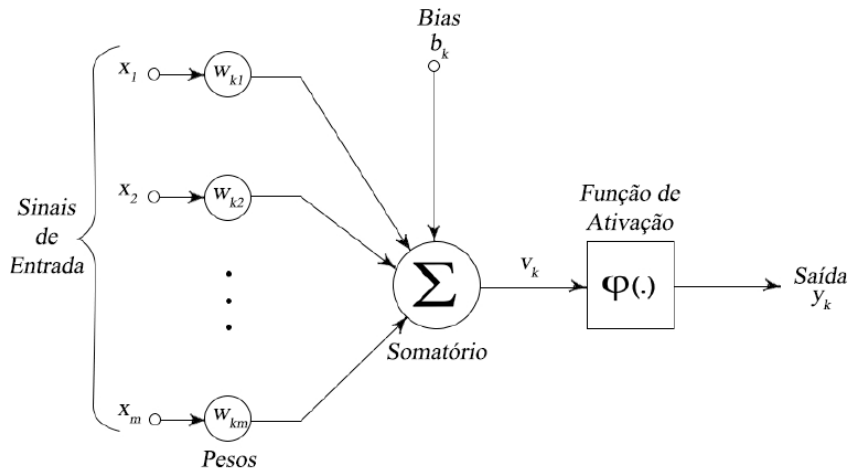


Figura (9) – Representação de um Neurônio Artificial (fonte: autor)

2.6.1.1 Funções de Ativação

Trata-se de uma função não linear cujo resultado é a saída do neurônio. Estas funções, em alguns casos, limitam a variação de saída do neurônio. Das funções de ativação mais comuns, a função Linear Retificada (ou **ReLU**) é uma das mais utilizadas devido à sua característica linear em certas regiões, o que torna mais fácil a computação dentro da rede. Na ReLU, o resultado do somatório dos produtos dos pesos pelas entradas é comparado com um limiar definido. Caso essa soma exceda o limiar, o neurônio transmite a

mesma informação, caso o contrário uma informação nula é transmitida. Matematicamente, a função ReLU é representada por:

$$ReLU(v_k) = \max(0, v) = \begin{cases} 0 & \text{se } v_k < 0 \\ v_k & \text{caso contrário.} \end{cases} \quad (2.17)$$

Outra função de ativação muito utilizada é a **Sigmóide**. Sua equação é dada por:

$$\sigma(v) = \frac{1}{1 + \exp(-v)} \quad (2.18)$$

A função Sigmóide, também chamada de função logística, apresenta uma não linearidade importante para redes neurais e é de fácil computação, uma vez que extrair sua derivada é fácil por se tratar de uma exponencial. Contudo, atualmente, ela é considerada obsoleta para uso em camadas internas de redes neurais; a função ReLU tem se mostrado melhor empiricamente para esta função. Todavia, a Sigmóide é adequada como função de ativação na camada de saída de problemas binários (GOLDBERG, 2017).

Na literatura, encontram-se ainda outras funções de ativação, tais como a **Softmax** (que é uma generalização da função sigmóide para problemas multiclasse) e a Tangente Hiperbólica **tanh**. Entretanto, por todas as vantagens citadas anteriormente e comprovadas pela comunidade, este trabalho utiliza as funções de ativação ReLU nas camadas internas e a função Sigmóide na camada de saída.

2.6.2 Perceptron Multicamadas

Uma rede Perceptron Multicamadas (*Multilayer Perceptron* - MLP) é um modelo de RNA muito utilizado em aplicações que exigem modelos supervisionados de aprendizagem. Ele possui, pelo menos, uma camada oculta, em que os sinais são propagados de forma direta pela rede, ou seja, não há propagação cíclica. Além disso, os neurônios desse modelo foram uma rede densa e estão amplamente conectados, ou seja, todos os neurônios de uma camada estão conectados com todos os neurônios das camadas adjacentes.

Quando uma MLP contém mais de uma camada oculta, ela deixa de ser uma RNA rasa e passa a ser uma Rede Neural Profunda, em que cada camada de neurônios realizam diferentes transformações nos dados (KAWAKAMI, 2008).

2.7 Aprendizagem Profunda

Bengio et al. (2009) afirma que, desde 2006, as redes neurais em *deep learning* vem sendo aplicada com sucesso nos mais distintos problemas, tais como classificação

(CHEN et al., 2014; SINGHANIA; RAO, 2017; B et al., 2018), regressão (QIU et al., 2014), problemas de NLP (COLLOBERT; WESTON, 2008), entre outros.

Na área de NLP, os modelos de *Deep Learning* vêm sendo cada vez mais utilizados para classificação de texto. Muitos desses modelos fazem classificação binária de notícias, atribuindo a elas um rótulo de falsa ou verdadeira. Esses modelos tem sido tão eficazes na detecção de notícias falsas que algumas abordagens recentes que utilizam deep learning, como Redes Neurais Convolucionais (CNN) e Redes Neurais Recorrentes (RNN) baseadas em memória de curto prazo (LSTM) para aprender representações textuais (YANG et al., 2016; JOHNSON; ZHANG, 2015; MA; GAO; WONG, 2018; JIN; CAO; GUO, 2017; B et al., 2018; SINGHANIA; RAO, 2017; LI; CAI; CHEN, 2018; MA et al., 2016a). Logo, motivado por estas abordagens e pelos resultados obtidos na revisão sistemática discutida na presente dissertação (ver Capítulo 3), serão analisadas mais detalhadamente as estruturas CNN e RNN sob o contexto do NLP.

2.7.1 Redes Neurais Convolucionais

As Redes Neurais Convolucionais, ou abreviadamente CNN, foram propostas no estudo de LeCun, Bengio et al. (1995) onde é proposta um tipo de *Perceptron* multicamadas que funciona de forma que os neurônios se conectam numa distribuição semelhante ao do córtex visual humano. Por conta disso, as CNNs foram inicialmente aplicadas para solução de problemas de computação visual e processamento de imagens.

Uma rede CNN consegue detectar padrões locais numa estrutura de dados, combinando-os afim de apresentá-los na forma de um vetor de tamanho fixo que representa a estrutura de entrada. Ela captura aspectos locais que contém maior informação afim de auxiliar na tarefa de predição de dados. A arquitetura convolucional possui uma vantagem em relação às demais que reside na sua rapidez em extrair características mesmo sem uma entrada de pesos inicial no treinamento. Outra vantagem das CNNs consiste na sua capacidade de prever comportamentos de padrões com componentes similares, mesmo que um desses padrões nunca tenham sido vistos durante o treinamento LeCun, Bengio et al. (1995).

2.7.1.1 Convolução 1D

A ideia principal por trás da CNN está na operação de convolução, seguida da operação de *pooling*. Uma função não linear é aplicada em uma janela de k palavras, que percorre toda a sentença. Esta função, comumente chamada de filtro, transforma uma subsequência de k palavras em um valor escalar. Considerando l diferentes filtros, tem-se um vetor l -dimensional, onde cada dimensão corresponde a um filtro de k -palavras. Desta forma, pode-se interpretar o tamanho do filtro como uma representação em *N-gram*.

Dada uma sequência de palavras $w_{1:n} = w_1, \dots, w_n$ que define uma sentença, Cada

palavra possui um vetor de características de dimensão d . Seja um filtro de tamanho k dado por um vetor de pesos $\mathbf{u}$. Considerando que o vetor de k palavras formado por este filtro na posição i , é dado por $\mathbf{x}_i = w_i, w_{i+1}, \dots, w_{i+k-1}$. Convoluir a sentença, significa, matematicamente, realizar o produto de Hadamard entre x_i e u , ou seja,

$$p_i = x_i \circ u \quad (2.19)$$

O resultado desta operação é um escalar a ser incluído na posição i do vetor de convolução. Um exemplo prático da Convolução 1D¹⁹ é representado na Figura 10. Ela mostra apenas o produto escalar da primeira componente, p_1 , do vetor de convolução. Uma vez que o filtro percorrer todas as componentes do trigramma “*Eu gosto deste*”, ele se deslocará para o próximo trigramma da sentença, no caso “*gosto deste filme*”.

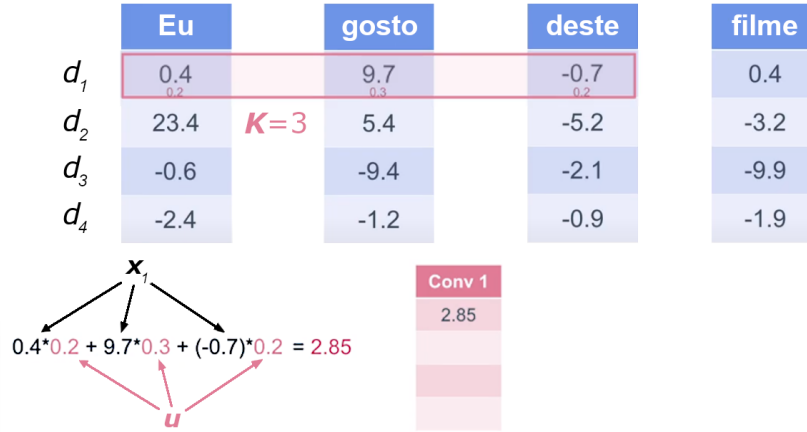


Figura (10) – Convolução 1D, com filtro $k = 3$ (fonte: autor)

2.7.1.2 Vetor de Pooling 1D

Aplicar a convolução sobre um texto, resulta em d vetores p_i , onde cada p_i de dimensão l , sendo l o número de posições diferentes do filtro no texto. Esta afirmação pode ser melhor visualizada na Figura 11. Observe que, no caso em questão, $l = 2$. Pois a sentença permitiu apenas 2 posições diferentes do filtro. A primeira coordenada de “Conv 1” e “Conv 2” é o vetor p_1 . Estes vetores são, então combinados (do inglês *pooled*) em um único vetor c que representa, idealmente, as características da sentença completa. A função de um vetor de pooling é para reduzir o tamanho da entrada afim de reduzir o número de parâmetros da rede. Contudo, essa redução, teoricamente, preserva as informações contidas na sentença.

A função de geração de cada coordenada do vetor de *pooling* é o que caracteriza seu tipo. Dentre eles, os mais comuns são o *Max-Pooling* e o *Global Max-Pooling*. O

¹⁹ O termo 1D é utilizado na literatura de NLP para evidenciar que trata-se de uma sequência horizontal de texto e não de imagens 2D

Max-Pooling é uma operação onde apenas o maior valor de cada coordenada dos vetores de convolução compõem o vetor de *pooling* e são levadas para as camadas internas da rede. A Figura 11 mostra este vetor sendo gerado através do *Max-Pooling*. O *Global Max-Pooling*, por sua vez, considera uma redução de dimensionalidade ainda mais drástica, pois nela apenas o valor máximo dentre todos os vetores de convolução é transmitido para as camadas internas.

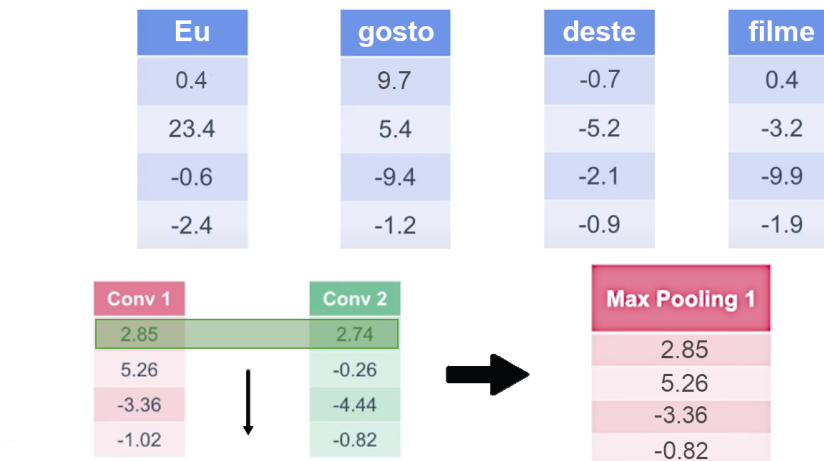


Figura (11) – Operação do *Max-Pooling* (fonte: autor)

2.7.2 Redes Neurais Recorrentes

Conforme explanado anteriormente, as CNNs permitem a codificação de uma sequência em um vetor de tamanho fixo. Contudo, apesar de manter o comportamento e a ordenação das palavras, esta análise se restringe a uma janela local. Logo, as CNNs na sua estrutura mais simples são boas em extrair informações de padrões locais mas desconsideram a ordem dos padrões que estão distantes na sequência do texto.

As RNN são uma solução para este problema, pois elas são adequadas para identificar padrões mais distantes ao longo do texto, sendo adequada para resolver problemas de longas sequências de dados (ZHANG; GHORBANI, 2019). Logo, tal como as CNNs, as RNNs apresentam soluções para problemas de NLP, tais como classificação de tópicos (COLLOBERT et al., 2011), análise de sentimentos (GLOROT; BORDES; BENGIO, 2011), resposta a perguntas (BORDES; CHOPRA; WESTON, 2014) e tradução de idiomas (JEAN et al., 2014; SUTSKEVER; VINYALS; LE, 2014).

Dentre os tipos de RNNs, destacam-se as LSTMs, que, curiosamente, costumam apresentar resultados melhores que sua versão padrão por resolver o problema da dissipação do gradiente (OSHIKAWA; QIAN; WANG, 2018a) e, finalmente, evitar o problema de dependência de longo prazo. Isso porque o comportamento natural das LSTMs é lembrar informações por longos períodos de tempo.

Diferentemente das redes MLP convencionais, as RNNs possuem uma certa quantidade de informação obtida através da realimentação das saídas de um ou mais neurônios. Sendo assim, os pesos dos neurônios são se alterando de acordo com as novas entradas e também ponderando seus valores através de estados passados, configurando no que se chama de *memória*. Na Figura 12, observam-se dois modelos de realimentação, no modelo da direita é feita a realimentação entre todos os neurônios; já no modelo à esquerda, cada neurônio é realimentado apenas com a informação dos demais.

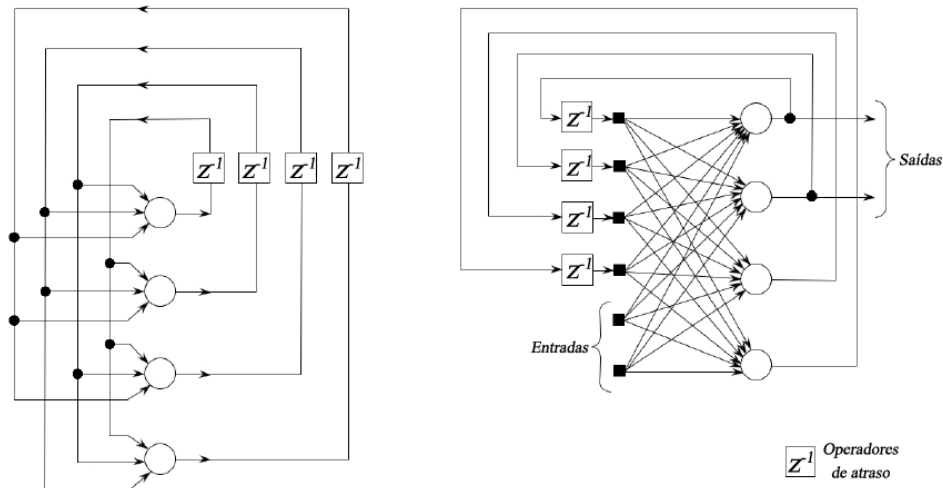


Figura (12) – Estrutura de uma RNN com modelo de realimentação Many-to-One (esquerda) e Many-to-Many (direita) (adaptado de (HAYKIN; NETWORK, 2004))

2.7.3 Long Short-Term Memory (LSTM)

As LSTM se caracterizam por serem uma RNN capaz de armazenar informações por longos períodos ao processar uma sequência temporal. Os pontos de memória de uma rede LSTM são denominadas células. Essas células são capazes de carregar informações até o final de uma sequência ou identificar as informações que devem ser esquecidas pela rede a partir de alguma etapa do processamento.

Hochreiter e Schmidhuber (1997) em seu estudo, propõe uma estrutura recorrente adaptada, juntamente com uma forma de recálculo dos gradientes obtidos no ajuste de peso através da *back propagation*. Essa estrutura faz com que a rede preserve uma maior quantidade de entradas passadas (*long-term memory*) sem abrir mão da relevância dos estados recentes. Para isso, a LSTM faz uso dos estados denominados como *hidden state* e *cell state* (Figura 13). mostra uma visão interna de um neurônio, chamado também de célula, de uma rede LSTM.

Observando atentamente a Figura 13 é possível notar que, num dado instante de tempo t , a célula possui como entradas o instante atual, x_t , o estado oculto h_{t-1} e o estado

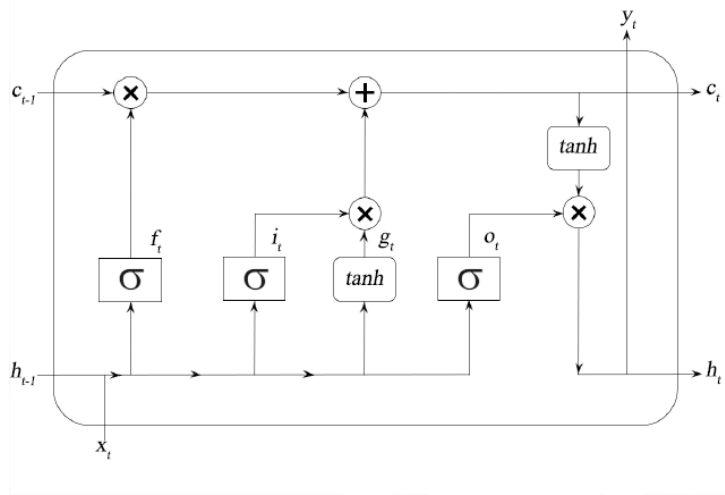


Figura (13) – Representação de uma célula LSTM (adaptado de (HOCHREITER; SCHMIDHUBER, 1997))

da célula c_{t-1} . Estes estados são recorrentes do instante passado $t - 1$, os atuais são c_t e h_t , bem como a saída y_t .

Além das entradas e saídas, uma célula LSTM é composta internamente de combinações de funções de ativação, adições e produtos. Estas operações são chamadas na literatura de *gates*, são elas, *forget gate*, *input gate*, *cell gate* e *output gate*. As equações que descrevem o funcionamento de cada uma delas são listadas abaixo:

$$\begin{aligned}
 f_t &= \sigma(x_t W_{i_f} + b_{i_f} + h_{(t-1)} W_{h_f} + b_{h_f}) \\
 i_t &= \sigma(x_t W_{i_i} + b_{i_i} + h_{(t-1)} W_{h_i} + b_{h_i}) \\
 g_t &= \tanh(x_t W_{i_g} + b_{i_g} + h_{(t-1)} W_{h_g} + b_{h_g}) \\
 c_t &= f_t \circ c_{(t-1)} + i_g \circ g_t \\
 o_t &= \sigma(x_t W_{o_i} + b_{o_i} + h_{(t-1)} W_{h_o} + b_{h_o}) \\
 h_t &= o_t \circ \tanh(c_t)
 \end{aligned} \tag{2.20}$$

Em linhas gerais, esse é o funcionamento de uma célula LSTM. Sua composição favorece para que seus estados internos propaguem a informação dos estados passado através da rede, de modo a dispensar uma alimentação realimentação explícita (como numa RNN tradicional) e, portanto, evitando o problema da dissipação do gradiente.

3 Revisão Sistemática

Uma Revisão Sistemática, ou abreviadamente RS, é um estudo que visa identificar e avaliar um tópico de pesquisa, reduzindo o conhecimento superficial e apresentando as lacunas e as melhores contribuições em uma área de pesquisa (PETERSEN et al., 2008; KITCHENHAM; CHARTERS, 2007). Uma RS passa por etapas de avaliação de qualidade, onde os estudos são analisados na sua profundidade, atestando sua consistência ou contradição (KITCHENHAM; BUDGEN; BRERETON, 2011).

A intenção desta RS é identificar as lacunas de pesquisa neste tema com o intuito de identificar oportunidades de pesquisa para esta presente dissertação.

Kitchenham e Charters (KITCHENHAM; CHARTERS, 2007) definem uma RS como um processo que consiste em três fases: Planejamento, Condução e Resultado. As fases de planejamento e condução deste estudo são descritas nas Seções a seguir, enquanto que a fase de resultados é apresentada na Seção 3.3.

3.1 Planejamento

3.1.1 Questões de Pesquisa

As questões de pesquisa para esta RS são:

RQ1: Como estão distribuídas as publicações sobre detecção de notícias falsas ao longo dos anos?

RQ2: Quais tipos de contribuições têm sido desenvolvidas no contexto da detecção de notícias falsas nas redes sociais?

RQ3: Quais são os resultados de detecção dos artigos mais qualificados?

RQ4: Quais algoritmos de Inteligência Artificial são aplicados nos estudos selecionados?

RQ5: Quais tipos de conteúdo são considerados para as soluções propostas?

3.1.2 Protocolo de Busca

Para obter os estudos que respondam às questões de pesquisa apresentadas, é necessário estruturá-las antes de construir um protocolo de busca. Isso pode ser feito por meio de uma tabela População, Intervenção, Comparação e Resultados “*Outcomes*”, ou abreviadamente PICO (HUANG; LIN; DEMNER-FUSHMAN, 2006).

O protocolo de busca foi definido simplesmente conectando os termos C_1 , C_2 e C_3 usando o termo lógico “AND” em todas as *keywords* presentes na Tabela 3.

Elemento	Descrição	Keywords	Referência
População	Sites de notícias, blogs e redes sociais, como Facebook e Twitter.	social network, social media, facebook, twitter, blogs, sites	C_1
Intervenção	Deteção de notícias falsas, desinformação boatos e rumores.	fake news, hoax, rumour, misinformation	C_2
Comparação	Comparar soluções usando IA das que usam sites de fact-checking	fact-check, fact-checking, machine learning	C_3
Resultados (“Outcomes”)	Verificar a efetividade dos resultados obtidos.	-	

Tabela (3) – Tabela PICO (*Population, Intervention, Comparison, and Outcomes*) para estruturação do Protocolo de Busca

Por fim, o protocolo de busca foi executado em cinco repositórios digitais, que possuem as publicações mais relevantes para o contexto deste trabalho, são eles:

- ACM Digital Library¹;
- IEEEExplore²;
- ScienceDirect³;
- Scopus⁴;
- SpringerLink⁵;

Como cada repositório possui sua própria sintaxe, a Tabela 4 mostra o protocolo de busca utilizado para cada motor de busca.

3.1.3 Critérios de Inclusão/Exclusão

Antes de seguir para a fase de Condução, é necessário definir os critérios de inclusão e exclusão a serem adotados nas seleções dos trabalhos, são eles:

¹ <http://dl.acm.org>

² <http://ieeexplore.ieee.org>

³ <http://www.sciencedirect.com>

⁴ <http://www.scopus.com>

⁵ <http://link.springer.com>

Tabela (4) – Protocolo de busca utilizados nos motores de busca

ACM Digital Library	"query": { (hoax OR misinformation OR fake news OR rumour) AND (social network OR social media OR social OR facebook OR twitter) AND (fact-check OR fact-checking OR machine learning) }
IEEEExplore	("Full Text & Metadata": hoax OR misinformation OR fake news OR rumour) AND ("Full Text & Metadata": fact-check OR fact-checking OR machine learning) AND ("Full Text & Metadata": social network OR social media OR social OR facebook OR twitter)
ScienceDirect	all:{(hoax OR misinformation OR fake news OR rumour) AND (fact-check OR fact-checking OR machine learning) AND (social network OR social media OR social OR facebook OR twitter)}
Scopus	TITLE-ABS-KEY((hoax OR misinformation OR fake news OR rumour) AND (fact-check OR fact-checking OR machine learning) AND (social network OR social media OR social OR facebook OR twitter))
SpringerLink	"search": {(hoax OR misinformation OR fake news OR rumour) AND (fact-check OR fact-checking OR machine learning) AND (social network OR social media OR social OR facebook OR twitter)}

3.1.3.1 Critérios de Inclusão

- Apresenta uma abordagem orientada à resultados de detecção ao invés de fazer análise de comportamento;
- Possui uma acurácia maior que 50%. Afinal uma classificação abaixo de 50% é pior do que aleatoriamente escolher se uma notícia é falsa ou verdadeira;

3.1.3.2 Critérios de Exclusão

- Trabalhos não escritos em inglês. Afinal, trabalhos qualificados acabam sendo publicados na sua versão em inglês. Até mesmo trabalhos qualificados em português possuem título e resumo escrito em inglês;
- Estudos duplicados;
- É sabido que a divulgação de notícias falsas se tornou mais frequente desde as eleições americanas de 2016 (ALLCOTT; GENTZKOW, 2017). Logo, esta RS excluirá todas as publicações feitas antes de 2016;
- Estudos fora do contexto;
- Apresenta uma solução sem uma análise comparativa;

- Possui um *dataset* para o experimento sem URL para download ou com menos de 1.000 notícias verdadeiras e falsas;
- Estudos que não atingiram 8 mais pontos quando submetidos aos critérios de qualidade definidos por esta RS;

3.1.3.3 Critérios de Qualidade

Para a análise de qualidade, os seguintes critérios foram designados para comparar os estudos selecionados. Foram elaborados 10 critérios de qualidade. Para cada critério, haverá 3 possíveis respostas: Não (escore=0), Parcialmente (escore=0,5), Sim (escore=1). Aqueles que atingirem nota 8 ou mais serão considerados aprovados na avaliação de qualidade desta RS. Eis os critérios:

Q01. As motivações e os objetivos de pesquisa estão claramente definidos?

Q02. Os métodos escolhidos são detalhados e justificados?

Q03. A solução proposta foi detalhada completamente?

Q04. O pesquisador especificou o dataset selecionado? E qual foi a justificativa para tal seleção?

Q05. O conjunto de dados foi analisado/interpretado para o experimento?

Q06. O estudo relata claramente os resultados?

Q07. As figuras e tabelas estão legíveis e bem desenhadas? O pesquisador discutiu cada uma delas?

Q08. Os resultados são consistentes com os objetivos apresentados?

Q09. As limitações e trabalhos futuros são apresentados neste estudo?

Q10. O artigo foi escrito de maneira clara e coerente?

3.2 Condução

Executando o protocolo de pesquisa considerando o texto completo para cada mecanismo de busca, conforme a Tabela 4, foram coletados 6.537 estudos primários. Vale salientar que esta pesquisa foi realizada ao longo do mês de Agosto de 2019. Portanto, trabalhos publicados posteriormente não foram considerados.

De toda forma, este montante é impraticável para se fazer uma RS. O primeiro passo, então, é fazer uma triagem dos títulos e *abstracts* pelas *keywords* mais relevantes

desta RS (STEINMACHER; CHAVES; GEROSA, 2013). Ainda assim, neste estágio, é possível que ainda continuem muitos falso-positivos (KITCHENHAM; CHARTERS, 2007).

A etapa de triagem trata de remover trabalhos duplicados e fazer uma rápida leitura de seus títulos e *abstracts*, buscando pelas *keywords* extraídas na Tabela 3. Para automatizar esse processo dispendioso, utilizou-se a ferramenta Rayyan⁶. Desta forma, 5.688 trabalhos foram excluídos pelos motivos listados na Figura 14, resultando em 66 estudos selecionados.

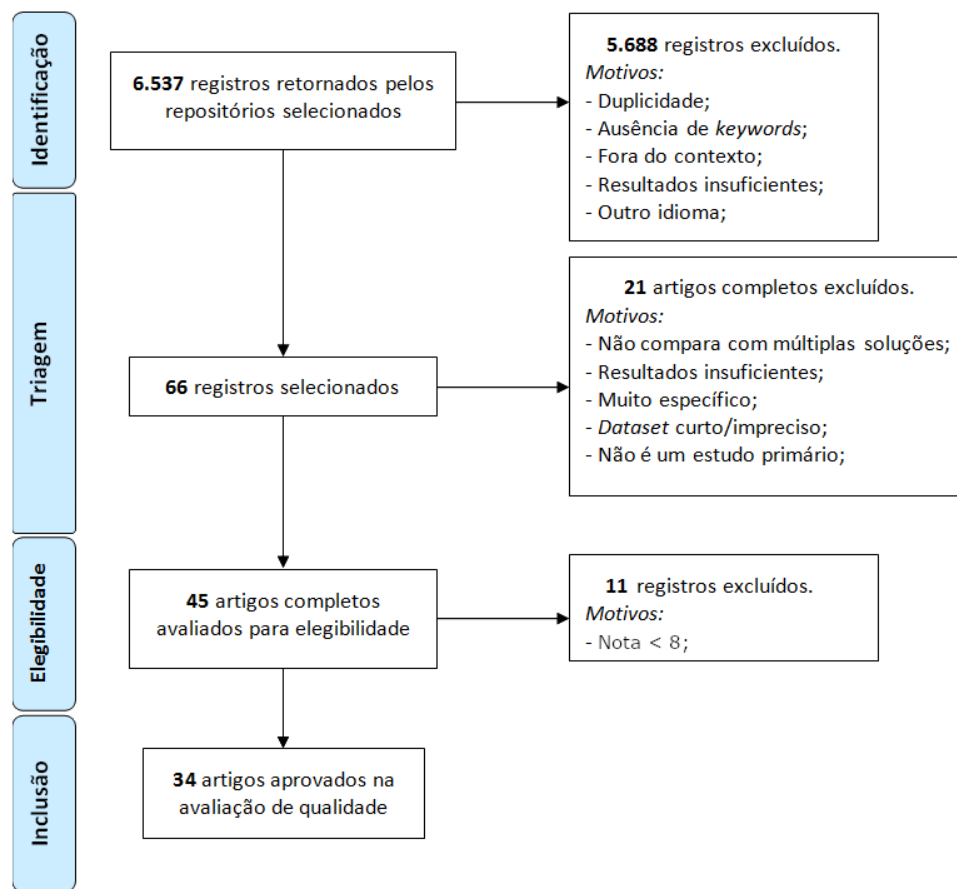


Figura (14) – Diagrama PRISMA adaptado para este trabalho.

Estes trabalhos seguiram para a etapa de elegibilidade, onde foram confrontados com todos os critérios de inclusão e exclusão, sobrando 45 estudos primários. Na última etapa de Condução, apresentada na Figura 14 (inclusão), os artigos são lidos por completo e submetidos aos critérios de qualidade, definidos por esta RS. Finalmente, esta RS selecionou os 34 trabalhos de maior qualidade acerca do problema de detecção de notícias falsas. Estes trabalhos respondem às questões de pesquisa elucidadas nesta RS a fim de gerar os resultados necessários.

⁶ <http://rayyan.qcri.org>

3.3 Resultados da RS

Nesta seção, todos os 34 estudos mais relevantes sobre o tema respondem às questões de pesquisa desta RS. Desta forma, será possível identificar o estado-da-arte da pesquisa e as oportunidades de pesquisas futuras.

3.3.1 RQ1: Como estão distribuídas as publicações sobre detecção de notícias falsas ao longo dos anos?

A Figura 15 reflete a intensidade de publicações que se propõem a resolver o problema de divulgação de notícias falsas. Esta distribuição sugere que em 2016, a pesquisa acerca deste tema era embrionária e pouco importante. É possível inferir que, após as eleições presidenciais americanas de 2016, a demanda por produção científica sobre este problema aumentou vertiginosamente.

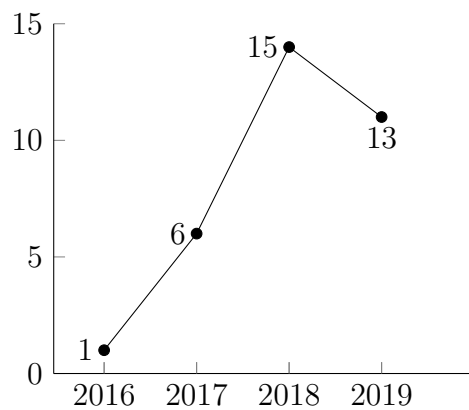


Figura (15) – Quantidade de estudos por ano.

Vale salientar que esta RS foi executada em Outubro de 2019. Logo, é possível que o número de publicações qualificadas em 2019 tenha excedido as de 2018.

3.3.2 RQ2: Quais tipos de contribuições têm sido desenvolvidos no contexto da detecção de notícias falsas nas redes sociais?

A Tabela 5 apresenta um panorama de quais tipos de estudos primários são mais publicados. Conforme mostrado na tabela, conferências são principal alvo dos trabalhos científicos, possuindo três vezes mais publicações que journals. Esta predominância pode ser justificada por conta das conferências possuírem uma natureza de publicação mais rígida e de escopo menor que *journals*. Além disso, nas conferências existem as discussões técnicas entre os pesquisadores nas sessões técnicas, enriquecendo mais o conhecimento acerca dos temas da conferência. Outro ponto a ser considerado é que o problema da detecção de notícias falsas é relativamente novo. Logo, seus trabalhos tendem a não esperar o processo mais demorado de revisão e aceitação de *journals* e livros.

Tabela (5) – Quantidade de publicações por tipo.

Tipo	Total	%
Book	3	8.8%
Conference	22	64.7%
Journal	9	26.5%

3.3.3 RQ3: Quais são os resultados de detecção dos artigos mais qualificados?

Para responder esta questão foi necessário fazer o mapeamento proposto na Tabela 6, onde os resultados de acurácia são valores que vão de 0 a 1. Como existem trabalhos que realizaram experimentos em mais de uma plataforma de rede social, será apresentado apenas o maior resultado de acurácia. Além da acurácia, esta tabela apresenta algumas informações que auxiliam nas respostas às demais questões de pesquisa, tais como o algoritmo de IA utilizado pelo autor para compor sua solução e a nota atribuída pela análise de qualidade.

Fazendo uma comparação puramente numérica dos resultados de acurácia de alguns trabalhos, é possível que se conclua, em um primeiro momento, que alguns estudos podem ser considerados quase que insatisfatórios pela disparidade de valores (como no estudo de (SHU; MAHUDESWARAN; LIU, 2018), por exemplo). Contudo, ao ler (SHU; MAHUDESWARAN; LIU, 2018), é possível encontrar uma considerável contribuição, além da acurácia. É importante tomar nota que, mesmo com resultados menores, existem outros fatores além da acurácia que podem motivar a escolha de um pesquisador ou desenvolvedor para adotar uma solução para um problema específico. Por exemplo, os resultados apresentados em (SHU; MAHUDESWARAN; LIU, 2018) superam os de outros estudos para as plataformas PolitiFact¹⁴ e BuzzFeed¹⁵. Adicionalmente, um estudo pode apresentar uma solução com uma acurácia menor, mas com bons resultados temporais.

Outra informação importante extraída a partir da Tabela 6 são os estudos de Monteiro, Santos e B (2018) e Silva et al. (2020), respectivamente **E33** e **E34**. A partir dos resultados desta RS, tem-se que o estudo de Monteiro, Santos e B (2018) foi o primeiro a abordar o problema da detecção de notícias falsas em língua portuguesa. Entretanto, sua principal contribuição reside na construção do *dataset* Fake.Br Corpus¹⁶, que será detalhado no Capítulo 5. Para testar o *dataset*, os autores fizeram um estudo de caso com alguns classificadores clássicos de aprendizagem de máquina, obtendo resultados que atenderam os requisitos da RS.

Já o trabalho de Silva et al. (2020), apesar de ter sido publicado posteriormente à execução da RS, continha em Outubro de 2019 uma versão *draft* revisada que foi considerada nesta revisão. Neste trabalho, os autores avançam em relação ao estudo de

¹⁴ <https://www.politifact.com/>

¹⁵ <https://www.buzzfeed.com/>

¹⁶ <https://github.com/roneysco/Fake.br-Corpus>

ID	Referência	Acc	Algoritmo	Nota
E1	(ZHOU et al., 2019)	0.980	CNN+GRU	9
E2	(YU et al., 2017)	0.933	CNN	8
E3	(CAI; BI; LIU, 2018)	0.902	LPT ⁷ + SVM	8
E4	(LI; CAI; CHEN, 2018)	0.889	Deep Bi-GRU	8
E5	(BHATTACHARJEE; TALUKDER; BALANTRAPU, 2018)	0.906	LR ⁸	8
E6	(YU et al., 2019)	0.948	Att-Based CNN	10
E7	(VEDOVA et al., 2018)	0.938	LR + HBLC ⁹	9
E8	(SHU; WANG, 2019)	0.878	NMF ¹⁰	10
E9	(B et al., 2018)	0.840	LSTM	9
E10	(RUCHANSKY, 2017)	0.953	LSTM	10
E11	(MA; GAO; WONG, 2017)	0.750	SVM	8
E12	(MA et al., 2016b)	0.896	SVM	9.5
E13	(ALKHODAIR et al., 2019)	0.780	LSTM	10
E14	(AHMED; TRAORE; SAAD, 2017)	0.920	SVM	8.5
E15	(WANG et al., 2018)	0.827	CNN	8.5
E16	(LIU; WU, 2018)	0.921	RNN + CNN	9.5
E17	(ZUBIAGA; LIAKATA; PROCTER, 2017)	0.691	CRF ¹¹ + MaxEnt	9
E18	(SHU; MAHUESWARAN; LIU, 2018)	0.742	RNN	9
E19	(LIU et al., 2018)	0.936	LR	10
E20	(XU; CHEN; MAO, 2018)	0.944	Att-Based LSTM	8.5
E21	(JIN; CAO; GUO, 2017)	0.788	Att-Based LSTM	8.5
E22	(VICARIO et al., 2018)	0.910	LR	8.5
E23	(ZHANG et al., 2019a)	0.809	Bi-LSTM + MLP ¹²	9
E24	(GENG et al., 2019)	0.956	Self-Att-GRU + BERT (DEVLIN et al., 2018)	10
E25	(MA; GAO; WONG, 2018)	0.737	Tree-structured RNN	8.5
E26	(GUO et al., 2018)	0.943	Bi-LSTM + HAN ¹³	8
E27	(KUMAR; SANGWAN; NAYYAR, 2019)	0.961	Decision Tree	9
E28	(SINGH; RANA; DWIVEDI, 2019)	0.805	2-Layer LSTM	8
E29	(AJAO; ZARGARI, 2019)	0.890	LSTM + HAN + SVM	9.5
E30	(REIS et al., 2019)	0.810	XGBoost	8
E31	(LIU; JIN; SHEN, 2019)	0.948	LSTM	8
E32	(WU; LIU, 2018)	0.912	LSTM	9
E33	(MONTEIRO; SANTOS; B, 2018)	0.890	SVM	10
E34	(SILVA et al., 2020)	0.965	LR	9.5

Tabela (6) – Resultados de detecção dos artigos selecionados

Monteiro, Santos e B (2018), reforçando a conclusão desta dissertação de que o trabalho de Monteiro, Santos e B (2018) foi o precursor acerca do problema para língua portuguesa. Silva et al. (2020) propõem uma solução de votação em conjunto (*ensemble*). Basicamente, os autores consideraram várias características isoladamente e compararam os resultados de detecção de cada característica para cada documento de entrada. A característica

que obtiver uma maior probabilidade é escolhida como a resposta oficial para aquele documento. Desta forma, por exemplo, se a solução considerando apenas as características textuais obtiver um resultado duvidoso em relação a um documento, as demais soluções (considerando outras características tais como, sentimentais ou semânticas) poderão ser utilizadas, caso obtenham uma maior probabilidade. Além disso, este estudo evolui em relação ao seu antecessor ao realizar ajuste de hiperparâmetro por meio de pesquisa em grade.

Contudo, os estudos de Monteiro, Santos e B (2018) e Silva et al. (2020) consideram resultados apenas na fase de treinamento e teste. Não é abordado um experimento considerando uma base de notícias desconhecida pelos modelos propostos pelos autores. Além disso, ambos os trabalhos não comparam com soluções de RNAs. Estas lacunas são exploradas na presente dissertação afim de proporcionar uma maior contribuição à comunidade científica acerca do tema.

3.3.4 RQ4: Quais algoritmos de Inteligência Artificial são aplicados nos estudos selecionados?

Analizando a Figura 16, pode-se perceber que a primeira metade (de HAN até GRU) compreendem as soluções usando *Deep Learning*; já a segunda metade (que vai de CRF até XGBoost) são soluções que utilizam algoritmos de *Machine Learning* para suas predições. Levando-se em consideração estes grupos, tem-se que 62.52% dos trabalhos usam algoritmos de *Deep Learning* em seus modelos de predição. Entretanto, o que mais chama atenção é que mais de 25% dos estudos usam LSTM para detecção. Isso se deve ao excelente comportamento desta rede neural para problemas de classificação de documentos.

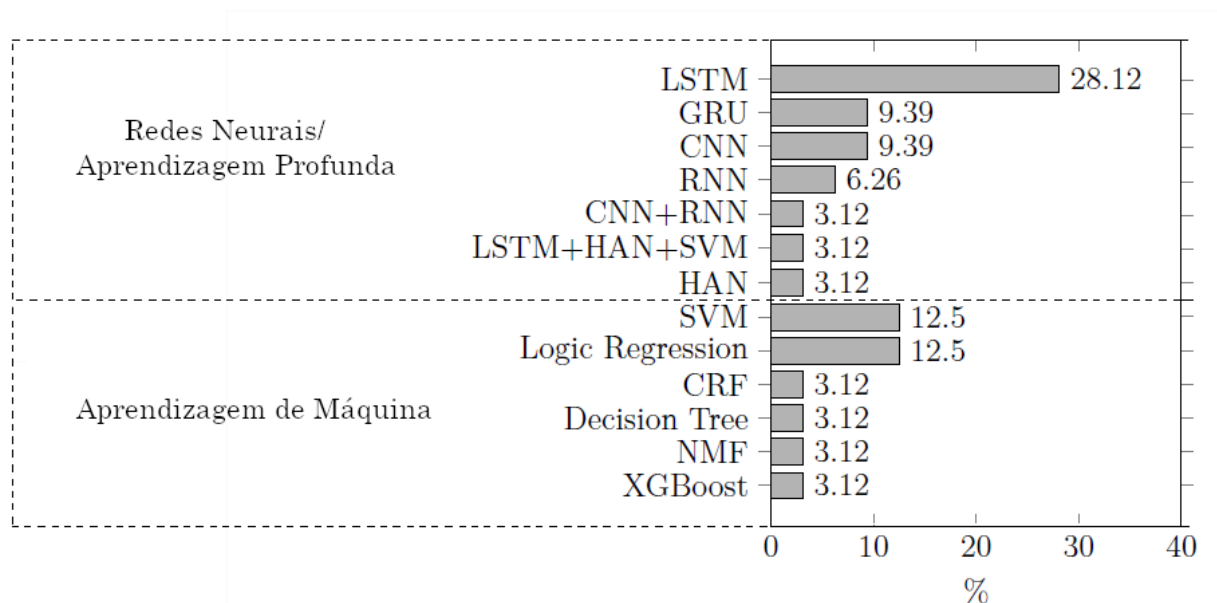


Figura (16) – Algoritmos de IA mais utilizados.

3.3.5 RQ5: Quais tipos de conteúdo são considerados para as soluções propostas?

Ao analisar a Figura 17, é possível ter uma visão geral das combinações de conteúdos considerados pelos trabalhos selecionados. Este gráfico mostra que os estudos mais relevantes ainda consideram o texto como o principal fator para identificar uma notícia falsa, corroborando com o que fora dito por (BONDIELLI; MARCELLONI, 2019a). Entretanto, a maioria dos estudos também mostram que, considerando aspectos sociais, é possível atingir melhores resultados de detecção.

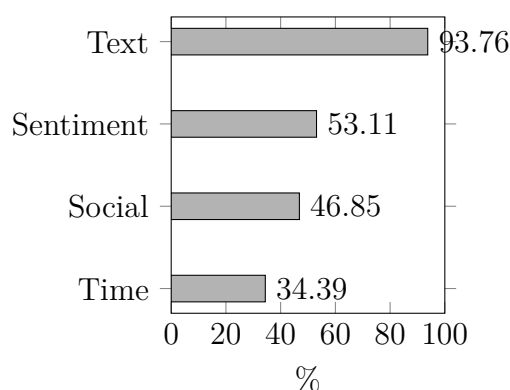


Figura (17) – Combinações de conteúdo consideradas.

3.4 Ameaças à Validade

Assim como em qualquer RS, o processo de condução e análise dos resultados deste trabalho pode conter falhas provenientes de vícios de análise. Outro ponto que cabe nota está relacionado aos critérios de qualidade, que podem não ter sido satisfatórios para conceituar um estudo como relevante, por conta de critérios subjetivos que dependem do perfil de cada pesquisador.

Além disso, muitos estudos são direcionados para detecção de rumores, que, apesar de sutis, possuem poucas diferenças das notícias falsas. Se esta diferença tivesse sido levada em consideração, esta RS teria retornado poucos resultados.

Um dos critérios de inclusão/exclusão foi o de considerar apenas trabalhos escritos em inglês. Isso, de certa forma, limita a contribuição de autores que, porventura, tenham escrito seus estudos em outros idiomas.

3.5 Conclusão da RS

A vantagem em se adotar um modelo completo de RS é que os artigos fora do contexto, que acidentalmente passaram pela fase de Triagem, podem ser eliminados nas

fases restantes antes da avaliação de qualidade. Isso reforça a importância de um processo completo de revisão sistemática da literatura.

O principal objetivo desta RS foi o de nortear esta dissertação acerca dos trabalhos mais relevantes sobre o tema e suas soluções. As respostas às questões de pesquisa mostraram que algoritmos de *Deep Learning* (em especial o LSTM) apresentam resultados de detecção expressivos. Finalmente, concluiu-se que considerar outros tipos de conteúdo, além da informação textual, como recursos sociais, temporais ou sentimentais, podem aprimorar a precisão das soluções propostas. Por fim, oportunidades de trabalhos futuros foram encontradas a partir dos resultados desta RS e serão discutidas no momento pertinente desta dissertação. Com relação à presente dissertação, considera-se que foi cumprido o objetivo **OE1** (ver 1.1).

Apesar das dificuldades, o problema da divulgação de notícias falsas precisa ser contido em qualquer idioma. Esta é outra motivação da presente dissertação: ampliar o debate científico acerca de técnicas de detecção de notícias falsas em português. Além do impacto inovador já mostrado nos resultados obtidos por esta RS, há também o impacto social de levar à sociedade ferramentas que mitiguem o problema da proliferação de notícias falsas na internet.

4 IDALina

Com base no levantamento teórico introduzido no Capítulo 2 e no estudo bibliográfico apresentado no Capítulo 3, é possível elaborar uma proposta de rede neural profunda baseada nas arquiteturas apresentadas nos estudos mais qualificados e adaptar este modelo para o problema da detecção de notícias falsas em português. A arquitetura proposta neste trabalho pode ser vista na Figura 18. Para tal propósito, a arquitetura conta com uma camada de redução de dimensionalidade por meio da representação por vetores de palavras embutidas (*Word Embeddings*) para cada uma das três características de entrada (semânticas, morfossintáticas e sentimentais). Em seguida, cada vetor de características passa por uma camada de convolução para que o modelo aprenda os padrões locais e posicionais para cada característica do texto. Estas informações são, por sua vez, enviadas a uma camada de concatenação, que realiza a junção e normalização das características do documento. Os dados são, então, inseridos na camada LSTM Bidirecional, que aprende o padrão sequencial das características em ambos os sentidos (*forward* e *backward*). Na penúltima camada, o modelo envia a informação para uma rede neural densa, que irá, basicamente, submeter os dados a mecanismos de normalização e regularização. Finalmente, na camada de saída, a rede profunda envia a resposta informando de se uma notícia de entrada é *Falsa* ou *Verdadeira*.

Como as camadas mais importantes são as camadas CNN e LSTM, onde o modelo, de fato, aprende as características do texto almejando a classificação correta dos dados de entrada, o nome atribuído à arquitetura foi IDALina - Uma proposta (I)nteligente de aprendizagem profunda para (D)etecção (A)utomática de notícias falsa na (Lín)gua Portugues(a). IDALina adota uma arquitetura híbrida, composta por CNN e LSTM, como pode ser observada na Figura 18. Cada camada da proposta é discutida mais detalhadamente ao longo deste capítulo.

4.0.1 Camada de Entrada

Esta camada é necessária para facilitar o entendimento do formato de entrada de dados.

Nesta etapa, são realizadas todas as técnicas de pré-processamento de texto expostas no Capítulo 2 afim de evitar padrões ruidosos que possam atrapalhar a aprendizagem da rede neural. Por fim, todos os documentos de treinamento, validação e teste são *tokenizados* e são atribuídos aos *tokens* uma representação no espaço vetorial. Desta forma, conclui-se que foi adotada a representação vetorial de *Bag-of-Words*, onde, cada palavra nova incluída no vocabulário do corpus recebe um valor numérico.

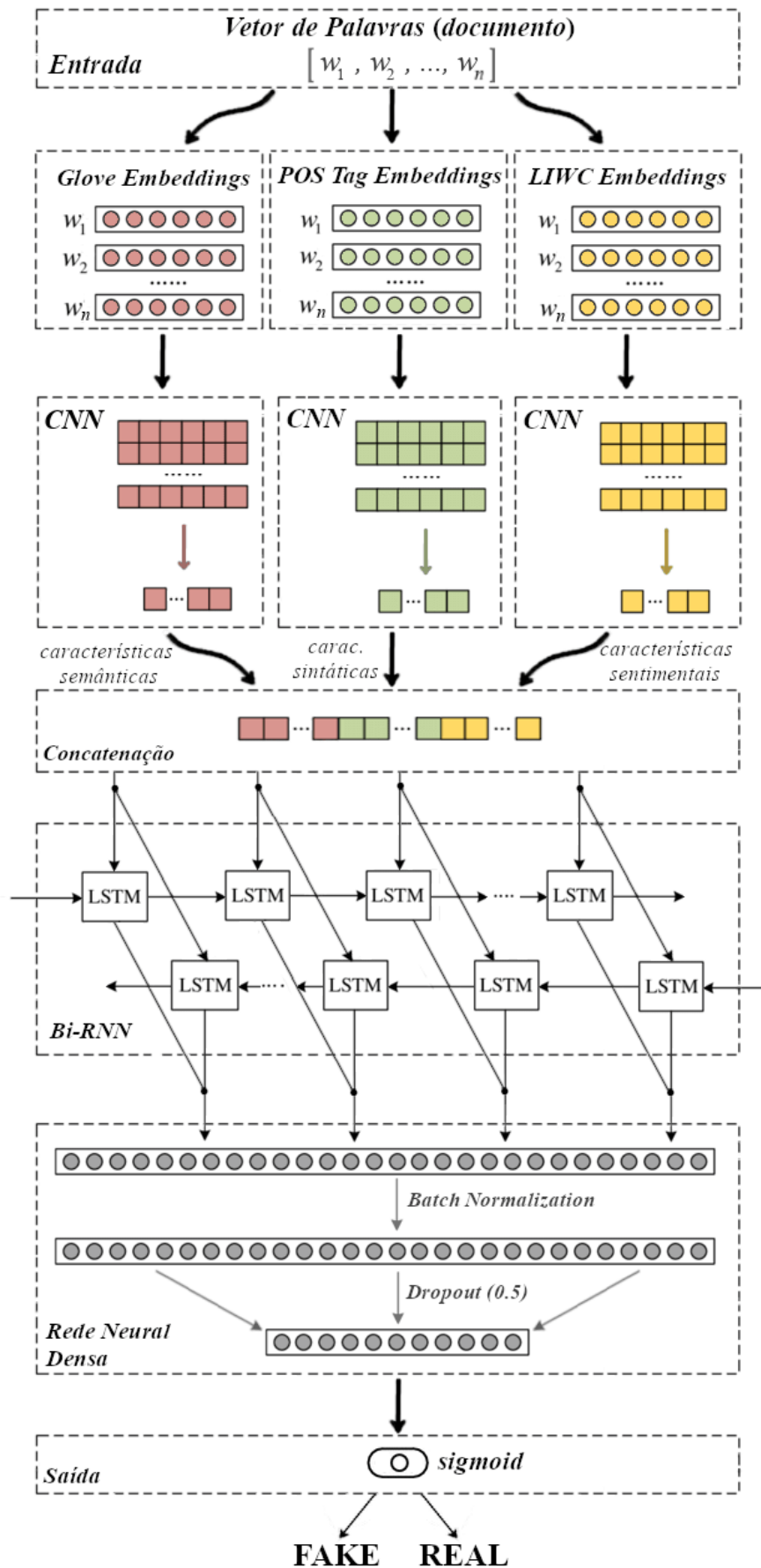


Figura (18) – IDALina.

4.0.2 Camada de *Embeddings*

Esta é a primeira camada da rede neural. Na literatura, alguns autores consideram a camada de *embedding* a real camada de entrada, pois nela os dados são representados levando em conta as características a serem aprendidas pela IDALina. No caso deste trabalho, as palavras são processadas na camada de *embeddings*, resultado em três tabelas representativas dos dados, uma para o espaço vetorial semântico, outra para o espaço vetorial morfo sintático e a última, para o espaço vetorial sentimental. Conceitualmente, pode-se dizer que as tabelas de saída da camada de *embedding* são como *lookup tables*.

Conforme citado no início deste trabalho, a IDALina considera características semânticas, morfo sintáticas e sentimentais (ou psicológicas). Na Figura 18, as características semânticas são geradas a partir do modelo GloVe (ver Seção 2.4.4). Já as características morfo sintáticas são obtidas através do etiquetador de Brill (ver 2.3.1) e as características sentimentais são geradas utilizando as classes da versão brasileira do LIWC (ver 2.5). Os motivos da escolha do etiquetador de Brill em detrimento dos demais elencados nesta dissertação, são discutidos na Seção 5.3.

4.0.3 Camadas CNN

A saída da camada de *embeddings* é composta de três matrizes tridimensionais, onde a primeira dimensão é o número de documentos do conjunto de treinamento, a segunda é o número de *tokens* e a terceira é composta pelas coordenadas do espaço vetorial de representação da palavra. Cada uma dessas matrizes representam uma das três características consideradas no problema.

O modelo, então, submete cada uma dessas matrizes, separadamente, a uma Rede Neural Convolutiva (CNN). A Figura 18 mostra que o modelo se divide em três ramificações. Esta divisão é necessária para que cada rede convolutiva se concentre em aprender os padrões de apenas uma característica e enviá-los às camadas mais internas da rede profunda.

Conforme detalhado em 2.7.1, uma rede convolutiva consiste na geração de uma matriz convolutiva seguida pela transformação dessa matriz em um vetor de *pooling*. Desta forma, cada rede CNN do modelo irá representar os dados não mais através de uma matriz tridimensional e sim por meio de uma matriz bidimensional, onde cada linha representa o comportamento de uma característica ao longo do documento.

4.0.4 Camada de Concatenação

Neste momento, o modelo possui três massas de dados distintas. Conforme supracitado, a camada convolutiva reduz a dimensão da entrada de dados representando em cada linha o comportamento de uma característica. Logo, cada coluna possui um

valor proveniente da transformação de combinação *pooling*. Desta maneira, esta camada trata simplesmente de concatenar as três características, mantendo a mesma referência do documento.

4.0.5 Camada Recorrente Bidirecional

Após a convolução e concatenação dos dados, faz-se a análise sequencial das distribuições. Para este tipo de análise, usam-se redes neurais recorrentes, em especial LSTM. Como a camada convolucional reduziu consideravelmente a massa de dados, esta rede não analisa sequencialmente as características de todas as palavras. Ou seja, ela irá aprender sequencialmente a distribuição resultante da convolução das três características.

Entretanto, a rede LSTM deve ser capaz de lidar com variações de valores provenientes do vetor de *pooling*. Consequentemente, apesar das palavras não estarem representadas sequencialmente diretamente, ainda há um certo nível de sequenciamento por conta do próprio mecanismo da convolução (lembrando que a convolução nada mais é do que percorrer uma janela, ou filtro, ao longo do texto e realizar o somatório de pesos e valores). Portanto, é importante uma análise de contexto de cada parâmetro de entrada em relação aos demais. Para evitar esse problema, a rede profunda deste trabalho adota a rede LSTM Bidirecional. Esta estrutura, consiste de duas LSTMs alimentadas pela mesma entrada, porém uma percorre do começo ao fim dos dados de entrada (*forward LSTM*) e a outra o faz no sentido inverso (*backward LSTM*). Nesta configuração a saída da LSTM Bidirecional (ou, abreviadamente Bi-LSTM) é composta por estas duas informações.

4.0.6 Camada Densamente Conectada

Neste ponto, a arquitetura recebe a saída da Bi-LSTM e passa por uma rede neural densamente conectada. Esta estrutura visa submeter a saída da LSTM a algumas subcamadas de correção dos dados, afim de evitar problemas como a dissipação do gradiente e o *overfitting*.

A primeira subcamada desta estrutura é a *Batch Normalization* (para mais detalhes de seu funcionamento, ver Anexo A.1.1). Em linhas gerais, ela trata de normalizar a saída da Bi-LSTM, acelerando o processo de treinamento, que resulta no aumento da taxa de aprendizagem e, por sua vez, a convergência do gradiente descendente.

Em seguida, IDALina aplica uma subcamada de regularização, chamada de *Dropout* (para mais detalhes, ver Anexo A.1.2. Em linhas gerais, trata-se de uma camada de regularização afim de evitar o *overfitting*, controlando a taxa de aprendizado para que ela não diminua muito rapidamente e prejudique o desempenho geral da rede.

4.0.7 Camada de Saída

Aplicadas as subcamadas de normalização e regularização, a IDALina está pronta para definir se a entrada inserida consiste de uma notícia falsa ou não.

Esta camada é constituída de apenas 1 neurônio com função de ativação *sigmóide*. Conforme discutido na Seção 2.6.1.1, a função de ativação *sigmóide* é apropriada para problemas de classificação binária. Isso se deve ao fato de que a sigmóide possui saída entre 0 e 1.

A saída de IDALina é tal que pode ser acoplada em qualquer aplicação web ou *desktop*. Basta que a mesma forneça um texto a ser tokenizado e consiga tratar uma saída 0 ou 1, onde 0 é uma notícia falsa e 1 é uma notícia verdadeira.

5 Experimentos e Análises

Uma vez definido o modelo que se propõe a satisfazer os objetivos traçados pelo autor, algumas decisões são necessárias antes de partir para os experimentos e análise de resultados, tais como a escolha da base de dados e das características de entrada para a IDALina. Além disso, é dedicada uma parte deste capítulo aos experimentos realizados ao longo deste projeto com o intuito de realizar o ajuste de hiperparâmetros. A motivação para este destaque é justificada pela importância estes parâmetros no desempenho da IDALina.

5.1 Escolha do *dataset*

A decisão acerca da base de dados a ser utilizada não exige muita análise. Afinal, sabe-se que existem poucas opções disponíveis e previamente rotuladas para o português até o presente momento da publicação deste trabalho. Com base na revisão sistemática feita pelo autor anteriormente, optou-se pela utilização da base de dados *Fake.Br Corpus*¹.

A escolha do *dataset* supracitado foi motivada por ser um banco de notícias completas e rotuladas apropriadamente para a detecção de notícias falsas. Além disso, este bando criado pelo grupo NILC-USP contém 7.200 notícias igualmente distribuídas para as duas classes do problema (3600 falsas e 3600 verdadeiras) dispostas em arquivos individuais de texto sem formatação. A janela de coleta deste corpus compreende notícias publicadas entre Janeiro de 2016 e Janeiro de 2018.

5.1.1 Análise do Corpus

Analisando o conteúdo deste corpus, é possível extrair informações importantes sobre seu comportamento. Na Tabela 7, percebe-se que notícias verdadeiras são maiores em tamanho em relação às falsas. Apesar de intuitivo, pode-se comprovar, por meio da análise de dados, a carência de conteúdo útil das notícias falsas. Nota-se também que o número de substantivos, adjetivos, advérbios e pronomes nas notícias verdadeiras é maior do que nas notícias falsas. Por outro lado, as notícias falsas, em geral, apresentam mais erros ortográficos em relação às verdadeiras.

¹ <https://github.com/roneysco/Fake.br-Corpus>

Features	True news	Fake news
Núm. médio de palavras	1268.5	216.1
Núm. médio de classes gramaticais (sem símbolos de pontuação e números)	494.1	119.2
Média de número de palavras (em caracteres)	4.8	4.8
Type-token ratio	0.47	0.68
Média de Núm. de Sentenças	54.8	12.7
Tamanho Médio das Sentenças (em palavras)	21.1	15.3
Núm. médio de verbos (normalizado)	13.4	14.3
Núm. médio de substantivos (normalizado)	24.6	24.5
Núm. médio de adjetivos (normalizado)	4.4	4.1
Núm. médio de advérbios (normalizado)	4.0	3.7
Núm. médio de pronomes (normalizado)	5.2	5.0
Núm. médio de stopwords (normalizado)	32.8	31.0
Percentual de textos com erros de ortografia	3.0	36.0

Tabela (7) – Análise do Corpus Fake.Br (MONTEIRO; SANTOS; B, 2018)

5.2 Desenvolvimento e Ferramentas Adotadas

Nesta seção são listadas as ferramentas e bibliotecas de programação utilizadas, tanto nos experimentos preliminares quanto nos experimentos que geraram os resultados desta dissertação. Tanto arquitetura da IDALina, proposta nesta dissertação, quanto os experimentos comparativos foram feitos em linguagem de programação *Python* na sua versão 3.7.1.

5.2.1 *Natural Language Toolkit* (NLTK)

O NLTK é um conjunto de módulos de código aberto para a linguagem Python. Foi desenvolvido em 2001, especialmente para trabalhar com processamento de linguagem natural. Esta biblioteca foi escolhida por várias universidades por ser uma ferramenta poderosa para o ensino e projetos de pesquisa. O NLTK foi desenvolvido com quatro objetivos: **Simplicidade, Consistência, Extensibilidade, modularidade**(NLTK, 2020). A versão do NLTK utilizada neste trabalho foi a 3.4.5.

5.2.2 Keras e Tensorflow

O Keras é uma biblioteca em Python que pode ser usada tanto isoladamente quanto como uma API *user-friendly* para as funcionalidades do TensorFlow. A sua grande vantagem é facilitar o processo de implementação do código de arquiteturas de *Deep Learning*. Em uma estrutura de código Keras existem dois tipos de API para criação de modelos, que são:

- *API Sequencial*: utilizada para arquiteturas de redes profundas onde suas camadas internas estão dispostas de forma linear.
- *API Funcional*: possibilita a criação de arquiteturas de rede mais complexas com topologias não-lineares, camadas compartilhadas e múltiplos canais de entrada e saída.

Como a arquitetura proposta neste trabalho se baseia na entrada de múltiplas características, escolheu-se usar a API funcional do Keras.

O TensorFlow é o *backend* padrão para o Keras. Ele é recomendado para redes envolvendo muitas camadas LSTM que necessitam do uso aceleração de GPU no hardware da Nvidia via CUDA e cuDNN. O TensorFlow também conta com classes de *callback* que monitoram todo o desempenho do treinamento da rede *learning rate*, auxiliando na análise da rede em tempo de treinamento. Este trabalho utilizou os recursos do Tensorflow de análise e atuação em tempo real durante os experimentos com a arquitetura proposta.

As versões do Keras e Tensorflow utilizadas nesta dissertação foram, respectivamente, 2.4.3 e 2.4.1.

5.3 Escolha do Etiquetador

Em sua camada de *embeddings*, a IDALina realiza transformações de redução de dimensionalidade em cima de três tipos de dados vetorizados. Um dos tipos de dados é resultante da etiquetagem morfosintática do texto, feito por um etiquetador. Existem vários etiquetadores disponíveis para o idioma português (alguns deles são apresentados em 2.3), o autor realizou um experimento comparativo afim de obter o etiquetador que melhor auxilia na tarefa de classificação.

Neste experimento, os etiquetadores foram submetidos a 3 classificadores: *Support Vector Machine* (SVM), Multi-Layer Perceptron (MLP) e uma rede neural LSTM simples. É importante destacar que este experimento se limitou apenas a comparar os etiquetadores sob o mesmo cenário. Por se tratar de um experimento comparativo, não foi intenção do autor obter o máximo desempenho em geral dos classificadores utilizados nesta análise. Logo, nenhuma tarefa de otimização foi feita, como ajuste de hiperparâmetros ou busca em grade.

Os classificadores SVM e MLP possuem as configurações *default* do NLTK. O classificador LSTM compreende uma rede simples com 10 unidades (esta quantidade de neurônios é a mesma definida para o classificador MLP), sem o uso de algoritmos de inicialização de pesos dos neurônios da rede, função de ativação recorrente *sigmóide* e função de ativação *tanh* (que são as funções *default* adotadas pelo Keras). Esta camada

se conecta diretamente à uma camada de saída composta por 1 neurônio que, através de uma função de ativação *sigmóide*, classifica as entradas de forma binária. Os resultados de performance são apresentados na Tabela 8.

Etiquetador	Modelo	Acc	P		R		F1-Score	
			Fake	True	Fake	True	Fake	True
LX-Tagger	SVM	.721	.72	.72	.73	.72	.72	.72
	MLP	.711	.71	.72	.72	.70	.72	.71
	VanillaLSTM	.740	.74	.74	.75	.74	.74	.74
HunPos	SVM	.749	.76	.74	.73	.77	.74	.76
	Random Forest	.753	.76	.75	.74	.77	.75	.76
	VanillaLSTM	.758	.78	.74	.73	.79	.75	.77
NLPNet	SVM	.781	.79	.77	.75	.79	.77	.78
	MLP	.761	.77	.75	.74	.78	.74	.76
	VanillaLSTM	.802	.82	.77	.78	.83	.80	.81
Brill	SVM	.778	.77	.79	.80	.76	.78	.77
	MLP	.816	.82	.82	.82	.82	.82	.82
	VanillaLSTM	.844	.84	.85	.85	.84	.84	.84

Tabela (8) – Performance de classificação considerando apenas as entradas dos etiquetadores

O experimento em questão mostra uma considerável vantagem do etiquetador de Brill treinado no corpus MacMorpho em relação aos demais, em especial nos classificadores baseados em Redes Neurais. Isso pode ser justificado devido ao modelo de Brill ser baseado em regras de transformação dirigidas por erro. Desta forma, ele tende a resolver melhor certas situações linguísticas que outros etiquetadores rotulariam erroneamente uma palavra, atingindo uma maior chance de acerto mesmo quando deparado com palavras desconhecidas.

Outra informação que pode ser interpretada analisando a Tabela 8 é a de que os etiquetadores LX-Tagger e a versão brasileira do HunPos cometeram mais erros no processo de extração de características morfossintáticas. Este dado mostra que tais etiquetadores poderiam não apresentar bons resultados nos experimentos finais por possuírem um Corpus orientado para o português europeu. Além disso, estes corpus possuem uma boa parcela de textos datados de 1380 até 1881, tornando, assim, mais difícil o acerto deste etiquetador quando submetido a uma sentença escrita em português contemporâneo.

5.4 Ajuste de Hiperparâmetros

Os modelos de redes neurais se diferenciam dos algoritmos tradicionais pela sua quantidade e, principalmente, complexidade de hiperparâmetros. Na aprendizagem profunda, este desafio no ajuste de hiperparâmetros é ainda maior, devido à quantidade de camadas internas (ZHANG et al., 2019b). Hiperparâmetros são variáveis que conduzem o processo de treinamento, influenciando fortemente nos resultados do modelo (GOLDBERG,

2017). Normalmente, estão associadas às características internas da rede profunda (embora, em alguns casos, pode estar associada à alguma característica do conjunto de entrada) (GOLDBERG, 2017; THANAKI, 2017).

Pode-se, então, encarar os hiperparâmetros como uma tarefa de decisão empírica, ou seja, deve-se alterar “à mão” e comparar o impacto no resultado que se quer otimizar (como, por exemplo, valor de perda, ou acurácia) (MIKOLOV et al., 2013; GOLDBERG, 2017).

5.4.1 Tamanho do Texto

Através da Tabela 7, percebe-se uma distância média acentuada entre o tamanho das notícias falsas e verdadeiras. Assim, caso nenhum tratamento em relação ao tamanho do texto fosse feito, seria possível que um algoritmo de aprendizagem de máquina (seja ele uma rede neural ou até mesmo um classificador clássico, como Naive Bayes, ou SVM) tendesse a classificar as notícias deste *dataset* simplesmente pelo tamanho delas. Ou seja, ele teria uma boa chance de acertar para notícias dentro da base de dados levando em conta mais o tamanho do que outras características.

Quando se pensa em preparação e normalização de dados, o que se quer é evitar que o modelo dê uma importância a certas características que, na verdade, estão desproporcionais ou não foram normalizadas pelo cientista de dados. Afinal, é necessário que o modelo consiga classificar corretamente inclusive casos onde, por exemplo, uma notícia falsa possui o tamanho médio de uma verdadeira. Desta forma, observa-se que o tamanho do texto a ser considerado como entrada para o modelo pode ser encarado como um hiperparâmetro.

Para escolher um valor que evite que a IDALina classifique de forma enviesada, foi feita uma análise da distribuição do tamanho dos textos sem “olhar” para o rótulo das notícias (ou seja, não importa se elas são falsas ou verdadeiras). Esta distribuição pode ser vista na Figura 19.

O diagrama da Figura 19 é chamado de Diagrama de Caixa (em inglês, *box and whisker plot*, ou simplesmente *boxplot*). No meio estatístico, é considerado um dos diagramas mais úteis, juntamente com o histograma, quando se quer estudar a distribuição dos dados. No diagrama de caixa, o conjunto de observação da variável é dividido em 4 partes iguais, chamadas quartis. O primeiro quartil, é o número que deixa 25% das observações abaixo e 75% acima; o terceiro quartil, deixa 75% das observações abaixo e 25% acima. Finalmente, o segundo quartil é a mediana, que deixa 50% das observações abaixo e 50% acima.

Inicialmente, foi realizado o experimento comparativo abordado na Subseção 5.3 utilizando o valor do terceiro quartil (que resulta em torno de 1000 *tokens* por documento). Este experimento foi executado nas seguintes especificações de plataforma:

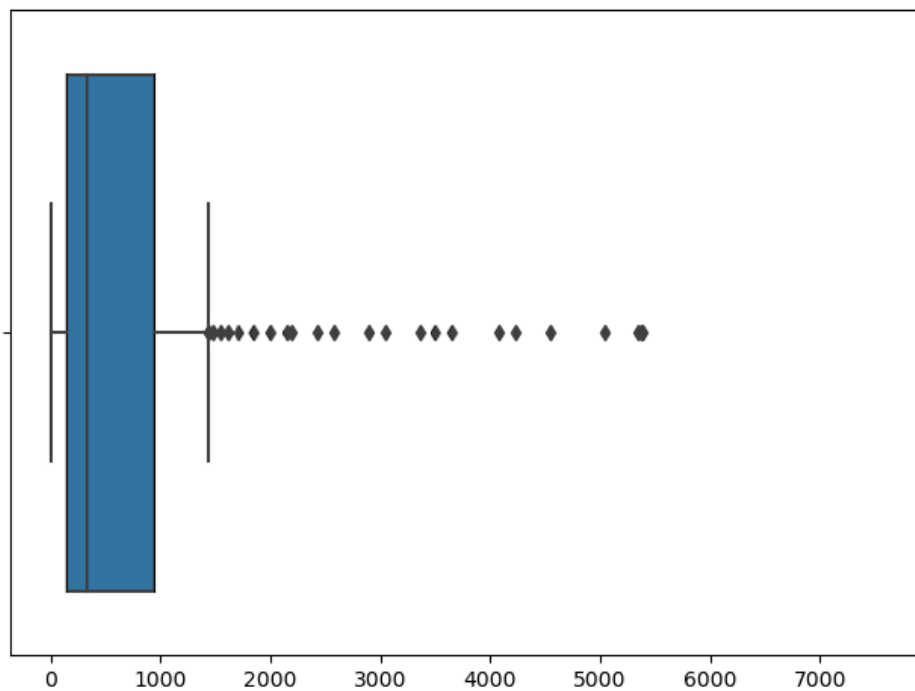


Figura (19) – Diagrama de caixa para o tamanho dos textos do *dataset*

- Sistema Operacional: Windows 8.1 Professional 64 Bits;
- Processador: Intel Core i5-3230M CPU 2.60GHz;
- Memória RAM: 16 GB
- Placa de Vídeo: AMD Radeon R9 M200X

Ocorre que este valor se mostrou inviável computacionalmente para os próximos experimentos deste trabalho. Com o recurso computacional apresentado acima, uma rede LSTM com 10 neurônios rodando em 5 épocas, cujos valores estão longe de serem considerados em comparação com a IDALina, que utiliza 128 células LSTM na subcamada de *forward* e outros 128 na camada de *backward*, levou um tempo médio de 2 horas por época. Logo, percebeu-se que considerar o valor do terceiro quantil não seria uma boa estratégia para os próximos experimentos.

Sendo assim, outra opção de medida seria a mediana. Entretanto, observa-se na Figura 19 que a mediana está muito próxima ao valor do primeiro quartil, o que faria com que muita informação do *dataset* fosse cortada. Então, resolveu-se adotar outra medida central de dispersão, chamada de desvio quartílico². O desvio quartílico nada mais é do que a média aritmética entre o limite inferior do primeiro quartil e o limite superior do terceiro quartil. Trata-se de uma medida de dispersão muito utilizada no ramo da estatística para definir a simetria de uma distribuição.

² <https://www.toppr.com/guides/business-mathematics-and-statistics/measures-of-central-tendency-and-dispersion/quartile-deviation/>

Aplicando esta métrica no hiperparâmetro de tamanho de texto, chega-se ao valor de aproximadamente 648 de desvio quartílico. Desta maneira, se considerar um valor de corte de 650, obtém-se o tamanho máximo da sequência de palavras adotado neste trabalho. Por conta desta análise ter sido feita após o experimento de comparação dos etiquetadores, o autor não realizou um novo experimento considerando agora o texto truncado em 650 palavras afim de comparar a redução do tempo por época de treinamento. Porém, é factível supor que o ganho de tempo seja em torno de metade do tempo para processar todo o corpus.

5.4.2 *Learning Rate*

Redes Neurais Profundas são treinadas usando um algoritmo, Gradiente Descendente Estocástico, do inglês *Stochastic Gradient Descent* ou abreviadamente SGD, (para mais detalhes do SGD, ver Anexo A.1). Neste algoritmo, um parâmetro fundamental para a boa convergência do modelo, auxiliando na obtenção da perda mínima geral, é a taxa de aprendizagem, do inglês *Learning Rate* ou, abreviadamente, LR.

Goodfellow, Bengio e Courville (2016) evidenciam, em seu livro, a importância da taxa de aprendizagem em relação aos demais hiperparâmetros:

“A taxa de aprendizado é, talvez, o hiperparâmetro mais importante. Se você tiver tempo para ajustar apenas um hiperparâmetro, escolha a taxa de aprendizado” (GOODFELLOW; BENGIO; COURVILLE, 2016, p. 429) (tradução nossa³).

Por conta disso, naturalmente, a taxa de aprendizagem é um hiperparâmetro.

5.4.2.1 Escolha do Otimizador

De acordo com os principais estudos encontrados na revisão sistemática apresentada no Capítulo 3, os otimizadores mais utilizados nos estudos selecionados são: o próprio SGD (com otimização de momentum), Adam e RMSprop. O otimizador SGD com momentum consiste em definir um valor percentual (de 0 a 1, embora estes extremos não sejam recomendados) para a atuação do otimizador no *learning rate*, elevando a taxa de aprendizagem e, com isso, acelerando a convergência do modelo em um estado ótimo (embora em alguns casos, esse aumento pode gerar a chamada explosão de gradiente).

Por sua vez, os otimizadores *Adam* e o *RMSprop* são, na verdade, versões otimizadas do SGD com controle de taxa de aprendizado adaptável ao contexto. Ambos se baseiam no cálculo do gradiente para tomar a decisão de mover o momentum, diferente do SGD cujo

³ The learning rate is perhaps the most important hyperparameter. If you have time to tune only one hyperparameter, tune the learning rate.

momento de atuação na taxa de aprendizado é definido antes do treinamento. O RMSprop atualiza a taxa de aprendizado através de uma função de decaimento que depende do quadrado da média dos gradientes. Já o otimizador Adam, faz este mesmo cálculo, porém ele mantém o histórico das taxas de aprendizado das épocas de treinamento anteriores que, por sua vez, influenciam na tomada de decisão da nova taxa de aprendizado para a iteração de treinamento seguinte. Estas funções de otimização, visam evitar que o gradiente exploda, fazendo o modelo divergir, gerando *overfitting*, bem como evitam que a taxa de aprendizado se mantenha constante, auxiliando o modelo a fugir de mínimos locais, evitando a estagnação em baixa performance, ou *underfitting*.

Baseado nestes conceitos, o autor realizou um experimento incluindo no modelo proposto os otimizadores SVG, RMSprop e Adam. Este experimento pode ser visto na Figura 20.

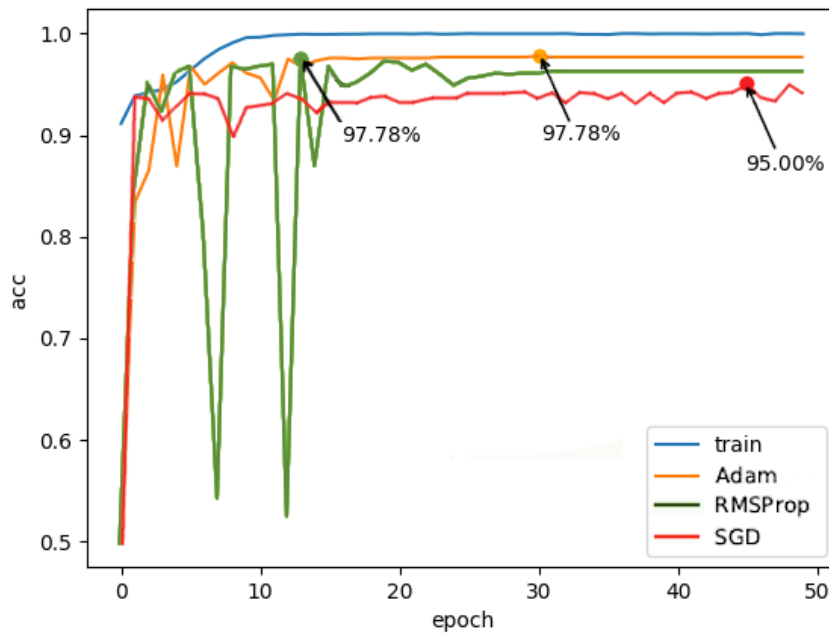


Figura (20) – Acurácia dos otimizadores ao longo do treinamento.

Pode-se concluir na Figura 20, que o otimizador mais indicado para o problema é o Adam nas suas configurações *default*. O otimizador SGD, como esperado, não obteve rendimento melhor que suas versões melhoradas. Nota-se também que o otimizador RMSprop, entre a 10ª e 15ª épocas, chegou a um pico em que atingiu a performance do Adam. Entretanto, a partir da 30ª época, quando o modelo parece convergir no seu limite, o otimizador Adam foi o único que se manteve com acurácia acima de 97%. Desta forma, o otimizador escolhido para o experimento final deste trabalho foi o Adam com uma taxa de aprendizado de η de 0.001.

5.4.3 Tamanho do Lote

É sabido que as redes neurais são treinadas usando um algoritmo de otimização chamado Gradiente Descendente Estocástico, independente de sua versão. Este processo envolve utilizar o estado atual para comparar as predições com os valores esperados e, a partir daí, extrair uma estimativa do erro do gradiente descendente e uma direção para onde os pesos dos neurônios devem seguir para fugir do mínimo local e buscar o mínimo ótimo.

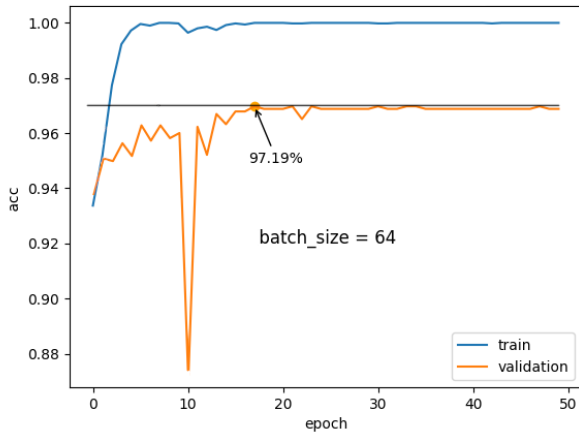
Como o gradiente descendente é uma medida estatística, quanto menos amostras de treinamento estiverem presentes neste cálculo, menos preciso é este valor da realidade, podendo gerar excessos de variação dos pesos na rede, dificultando a taxa de aprendizagem e, por sua vez, a convergência do modelo.

Por outro lado, quanto mais amostras, maior é o custo computacional. Além disso, alguns problemas apresentam um gradiente descendente mais “ruidoso”, fruto da existência de mínimos locais resultantes da complexa separação das classes do problema. Nestes casos, um tamanho maior de amostras presentes no cálculo do gradiente também dificultará a convergência do modelo para uma performance ótima no processo de aprendizagem (em alguns casos, o modelo poderá até divergir).

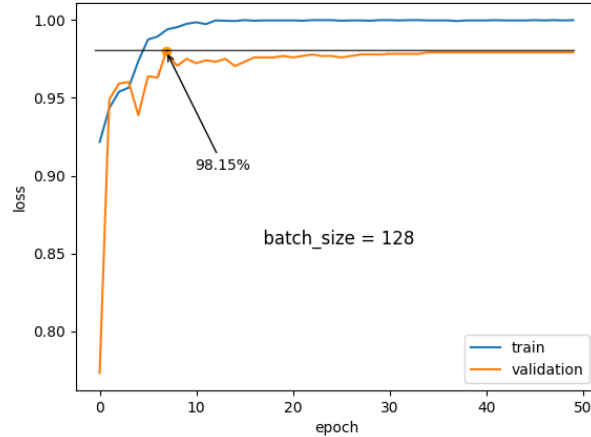
Logo, o tamanho de amostras consideradas no cálculo do erro do gradiente é outro hiperparâmetro de uma rede neural profunda, chamado de tamanho do lote. Um lote, ou *batch*, de 32, por exemplo, significa que foram escolhidas, aleatoriamente, 32 amostras do conjunto de treinamento para estimar o gradiente descendente antes de alterar os pesos de toda a rede. A partir daí, surge um conceito muito utilizado em redes neurais profundas, chamado época, ou *epoch*. Uma época de treinamento significa que a rede passou por todos os lotes de treinamento e está pronta para atualizar os pesos dos neurônios para uma próxima época. Conceitualmente, quando o tamanho do lote é entre 1 e o tamanho do treinamento, diz-se que está sendo adotada a otimização do Gradiente Descendente em Mini-Lotes.

Feitas estas considerações, foi realizado um experimento de análise do impacto do tamanho do lote na performance do modelo proposto. Este experimento está retratado na Figura 21. Foram escolhidos os tamanhos de lote de treinamento de 64, 128, 256 e 512. Esta escolha foi inspirada nas recomendações do livro de Goodfellow, Bengio e Courville (2016), que afirma ser uma boa prática começar com o tamanho de lote de 32 ou 64 e aumentar (ou diminuir) em potências de base 2. Os autores do livro justificam esta recomendação apontando a melhora significativa no processamento das GPUs quando os lotes possuem estes tamanhos.

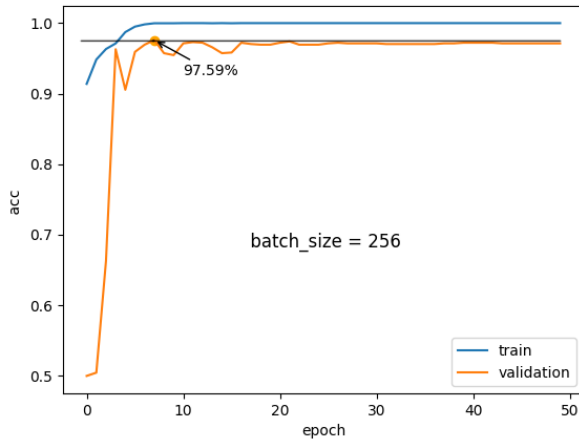
Analisando as performances dos tamanhos de lote, a Figura 21a confirma que um tamanho do lote menor, de fato, proporciona uma taxa de aprendizado menor e mais



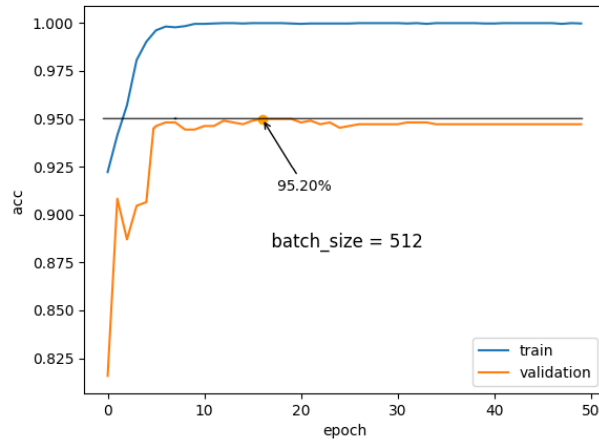
(a) lote de 64



(b) lote de 128



(c) lote de 256



(d) lote de 512

Figura (21) – Impacto do tamanho do lote na performance do modelo.

ruidosa. Por outro lado, a Figura 21d mostra que um tamanho de lote muito grande, gera perda de performance e estagnação do modelo. A partir deste tamanho, conforme Keskar et al. (2016) concluem em seu estudo, o aumento no tamanho do lote iria desencadear uma degradação do modelo e uma perda de capacidade de generalização. Logo, a escolha do tamanho de lote mais adequada seria 128 ou 256.

Comparando o comportamento do modelo com o lote de 128 (Figura 21b) e com o de 256 (Figura 21c), encontra-se que ambos convergem após a metade do experimento. Contudo, o lote de 128 parece ter descido mais no gradiente, atingindo um valor de mínimo melhor que o de 256, proporcionando uma performance sensivelmente melhor.

5.4.4 Demais Parâmetros

Além dos experimentos já discutidos neste capítulo, o autor realizou, no âmbito deste trabalho, outros experimentos de análise de performance, tais como o impacto do

tamanho do filtro nas camadas CNN e o percentual do *dropout*. Nestes casos, a alteração destes não provocou uma queda ou um aumento substancial no processo de treinamento e avaliação do modelo. Estes comportamentos já eram esperados com base nos experimentos de Zhang e Wallace (2015). Em seu estudo, Zhang e Wallace (2015) analisa o impacto de mudanças nos parâmetros das redes convolucionais na classificação de documentos, dentre elas o tamanho do filtro e o percentual de *dropout* em uma rede densa conectada à saída. Ele conclui que, para redes CNN de filtro simples (caso deste trabalho) percebe-se uma variação de menos de 0,5% para mais ou para menos de acurácia ao mudar o tamanho do filtro e o *dropout*. Desta forma, optou-se por discutir mais detalhadamente a taxa de aprendizagem e o tamanho do lote que, de acordo com a literatura, são os hiperparâmetros mais importantes de uma rede neural profunda, pois eles conduzem todo o processo de treinamento da rede.

5.5 Resultados

Esta seção discorre sobre o processo de treinamento, obtenção e análise dos resultados. Dentre os objetivos listados por este trabalho na Subseção 1.1, está o de superar os trabalhos de Monteiro, Santos e B (2018) e Silva et al. (2020), que são tidos como o estado-da-arte sobre o tema para o idioma Português.

5.5.1 Processo de Treinamento, Validação e Teste

Para o experimento final, foi utilizado um método de divisão de dados baseado em três conjuntos: treinamento, validação e teste. No caso, 70% do *dataset* pertencem ao conjunto de treinamento, 15% ao conjunto de validação e os 15% restantes ao conjunto de teste. As amostras destes três subconjuntos do *dataset* são escolhidas aleatoriamente mantendo a igual distribuição das classes. Ou seja, metade de cada subconjunto é composta por notícias rotuladas como verdadeiras e a outra metade é composta por notícias falsas. Neste método, o conjunto de teste é separado e não faz parte do experimento de treinamento e validação. Logo, considera-se que este conjunto está fora da base de acesso ao modelo e faz parte de um experimento à parte, chamado de experimento de campo.

Desta forma, os conjuntos de treinamento e validação são os únicos enviados à IDALina, que inicia uma *época* dividindo a base de treinamento em lotes de 128 amostras, conforme discutido na seção de hiperparâmetros. Então, o modelo realiza as iterações necessárias para percorrer todos os lotes até atingir o momento da validação da época atual. Nesta etapa, a rede acessa o conjunto de validação para avaliar se ele está convergindo ou não para um estado ótimo. Ao final de todas as 50 épocas executadas neste experimento, caso o modelo convirja, é armazenada a configuração da rede com os melhores resultados no conjunto de validação. Considera-se que, neste ponto, o modelo está treinado e suas

informações de classificação (acurácia, *precision*, *recall* e *F-measure*) servem de base comparativa em relação aos demais algoritmos.

5.5.2 IDALina *versus* RNAs Clássicas

Neste experimento, o modelo proposto por esta dissertação é confrontado com arquiteturas *shallow*⁴. A Figura 22 mostra que o modelo proposto por este trabalho apresenta uma acurácia maior que os demais durante, praticamente, todo o treinamento. Além disso, observa-se que, por volta da 30^a época, o modelo converge para um valor muito próximo da acurácia máxima limitada pela linha horizontal preta.

Este gráfico também mostra que, isoladas, as redes CNN e LSTM não obtiveram um desempenho melhor que o modelo proposto. Isso pode ser explicado pelas características de cada estrutura. As LSTMs são boas em identificar o sequenciamento de características, mas, normalmente, têm dificuldade em encontrar características locais em um documento. Além disso, são computacionalmente custosas e tendem a entrar em *overfitting* quando atingem uma quantidade de épocas suficientes para que elas memorizem boa parte da sequência de entrada. Com as CNNs, ocorre o oposto. Elas têm facilidade em evidenciar características locais relevantes do texto e são computacionalmente rápidas ao passo que possuem dificuldade no sequenciamento da entrada por atenuar este sequenciamento já na matriz de convolução.

Por conta disso, o modelo proposto supera as demais redes por possuir as vantagens de ambas e mitigar, por meio de uma camada densa de normalização e regularização, as desvantagens de ambas.

Outro dado importante que pode ser encontrado na Figura 22 é o rendimento da rede GRU superior ao LSTM. É sabida que a única diferença comprovada da GRU em relação à LSTM reside no menor tempo de computação dos *gates* e das funções de ativação. Contudo, alguns estudos mostram que a GRU pode apresentar melhores resultados em alguns problemas (KHANDELWAL; LECOUTEUX; BESACIER, 2016; ZULQARNAIN et al., 2020; YIN et al., 2017). Portanto, é importante considerar, futuramente, a hipótese de adaptação do modelo proposto para usar células GRUs ao invés de LSTMs.

5.5.3 IDALina *versus* Demais Abordagens

Apesar do experimento anterior mostrar o potencial da arquitetura IDALina, o leitor ainda pode considerar pouco vantajoso usá-la para obter um ganho de acurácia de pouco menos de 2%. Entretanto, o modelo proposto supera outras medidas de performance. Para isso, foi feito o mesmo experimento com as mesmas redes neurais básicas do experimento

⁴ Rede *shallow* é um termo que se dá na literatura para uma rede na sua versão mais simples sem ajustes de hiperparâmetros.

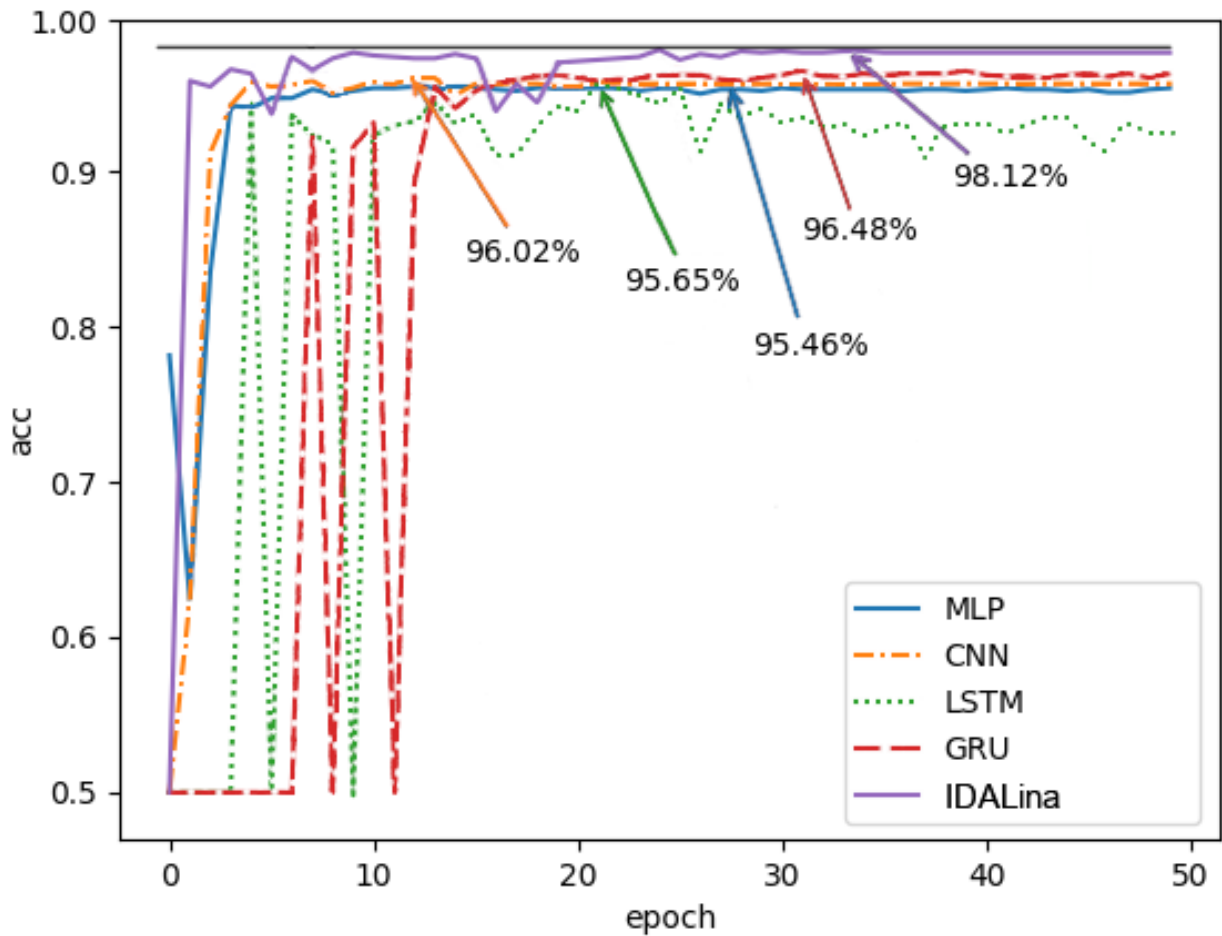


Figura (22) – Performance do Modelo Proposto (cor roxa) em relação à demais RNAs

anterior, porém, agora considerando todas as métricas de performance. Além disso, foram incluídas as abordagens utilizando classificadores de aprendizagem de máquina nos estudos de Monteiro, Santos e B (2018), Silva et al. (2020) e, com isso, satisfazer o Objetivo **OE2** desta dissertação (ver Subseção 1.1).

5.5.3.1 Medidas de Performance

Para comparar os resultados, foram adotadas as mesmas medidas de performance descritas por Monteiro, Santos e B (2018) e Silva et al. (2020), uma vez que este trabalho realiza a comparação com os resultados de ambos os estudos. São elas:

- *False Positive Rate* (FPR): consiste na proporção de notícias falsas erroneamente classificadas como verdadeiras. Logo, quanto menor esta medida, melhor é o classificador;
- *Recall* (R): trata-se da razão entre o total de verdadeiros positivos em relação ao total que realmente é positivo (portanto, quanto maior, melhor). Pode ser representada através da seguinte fórmula $R = \frac{TP}{TP+FN}$;

- *Precision* (P): dada pela proporção o total de verdadeiros positivos em relação ao total classificado como positivo (logo, quanto maior, melhor). Pode ser representada através da seguinte fórmula $R = \frac{TP}{TP+FP}$;
- *F-measure*: é obtida através da média harmônica entre o *precision* e o *recall*.

	FPR	R	P	F1-measure
MLP	0.045	0.955	0.955	0.955
CNN	0.040	0.960	0.960	0.960
LSTM	0.043	0.957	0.957	0.957
GRU	0.036	0.964	0.964	0.964
(MONTEIRO; SANTOS; B, 2018)	-	0.890	0.880	0.890
(SILVA et al., 2020)	0.024	0.954	0.976	0.965
IDALina	0.019	0.981	0.981	0.981

Tabela (9) – Comparação entre os resultados do modelo proposto e as demais abordagens.

Na Tabela 9, observa-se que a IDALina supera tanto os classificadores baseados em RNAs quanto as soluções abordadas pelos estudos de Monteiro, Santos e B (2018) e Silva et al. (2020) em todas as medidas de performance. Destaca-se para a diferença de quase o dobro de taxa de falso positivo da IDALina em relação à rede GRU (para as demais RNAs esta diferença passa do dobro); já em relação ao trabalho de Silva et al. (2020), a IDALina apresenta uma redução de 25% de taxa de falso negativo.

À princípio, estes números parecem pequenos. Porém, considere, por exemplo, que um destes classificadores seja incorporado a um software de detecção automática que será vendido para uma rede social. Imagine que ele fará parte de uma solução para auxiliar no disparo de bloqueio de usuários disseminadores de notícias falsas. Este classificador será submetido a uma carga massiva de bilhões de dados a serem analisados (afinal, é uma rede social). Portanto, quando se fala de uma quantidade grande de entrada de dados, um percentual pequeno se torna mais relevante e impactante para problemas deste tipo.

5.5.3.2 Contribuições deste Trabalho

Neste tópico são apresentadas as contribuições da presente dissertação em relação aos demais estudos relacionados. Para esta breve análise comparativa, é preciso listar algumas questões de pesquisa:

- **O estudo considera Características Não-Textuais? (Q1):** define se foram exploradas características além das textuais, tais como características sintáticas, semânticas e emotivas;
- **O estudo contorna o problema da dimensionalidade na representação vetorial de palavras? (Q2):** verifica se o estudo busca outra forma de representação

vetorial que não é baseada em contagem (como *one-hot encoding* ou TF-IDF). Trata-se de uma questão de pesquisa importante para dar à solução proposta mais escalabilidade em caso de ser submetida a um corpus de vocabulário muito grande.

- **A pesquisa compara a solução com classificadores de *machine learning* e redes neurais? (Q3):** esta questão de pesquisa é importante para dar mais robustez ao trabalho analisado;
- **O autor se preocupa em realizar ajustes de hiperparâmetros? (Q4):** estudos que consideram o ajuste de hiperparâmetros apresentam mais confiabilidade em seus resultados de desempenho por extrair o melhor possível de suas soluções;
- **A pesquisa realiza testes fora da base de treinamento e validação (Q5):** muitos estudos apresentam resultados obtidos na etapa de validação, mas nem todos realizam testes fora da base de treinamento e validação para verificar se o nível de queda de performance, que é algo normal e perfeitamente esperado, é aceitável.

Com base nestas questões, a Tabela 10 compara as contribuições desta dissertação com os trabalhos relacionados. Na tabela, considera-se que, se um estudo satisfaz aquela questão, então é marcado com um “x” na coluna da respectiva questão.

	Q1	Q2	Q3	Q4	Q5
(MONTEIRO; SANTOS; B, 2018)	x				
(SILVA et al., 2020)	x	x		x	
IDALina	x	x	x	x	x

Tabela (10) – Comparativo das contribuições desta dissertação com os trabalhos relacionados.

Este trabalho se destaca em relação aos demais, especialmente na parte de experimentação. Isso pode ser visto na Tabela 10, que mostra que, além de apresentar melhores resultados em todas as medidas de performance, este trabalho realiza comparações com mais tipos de arquiteturas distintas e faz experimentos com uma base externa à de treinamento e validação.

5.5.4 Desempenho do Modelo fora da Base de Treinamento e Validação

É importante destacar que os estudos de Monteiro, Santos e B (2018) e Silva et al. (2020) adotaram a abordagem de validação cruzada de todo o *dataset* em 5 dobras (*5-fold cross-validation*) para as etapas de treinamento e validação. Contudo, apesar de ser uma técnica consagrada na literatura, ela não realiza o tipo de teste abordado neste tópico.

Este experimento visa abordar um cenário mais prático, onde a rede profunda é modelada, treinada, colhida suas medidas de performance, mas são submetidas à outra base de notícias nunca vistas pelo modelo nem na fase de validação. Tratam-se do conjunto

que compreende 15% do *dataset* e foi separado no antes de todos os experimentos deste trabalho. Sendo, portanto, desconhecido pelo modelo.

Com isso, foi possível avaliar as predições reais e verificar se o modelo sofrerá uma queda de desempenho muito acentuada ou se ele irá se manter próximo em relação às medidas de desempenho da etapa de validação. A Figura 23 apresenta o desempenho do modelo submetido à base de teste.

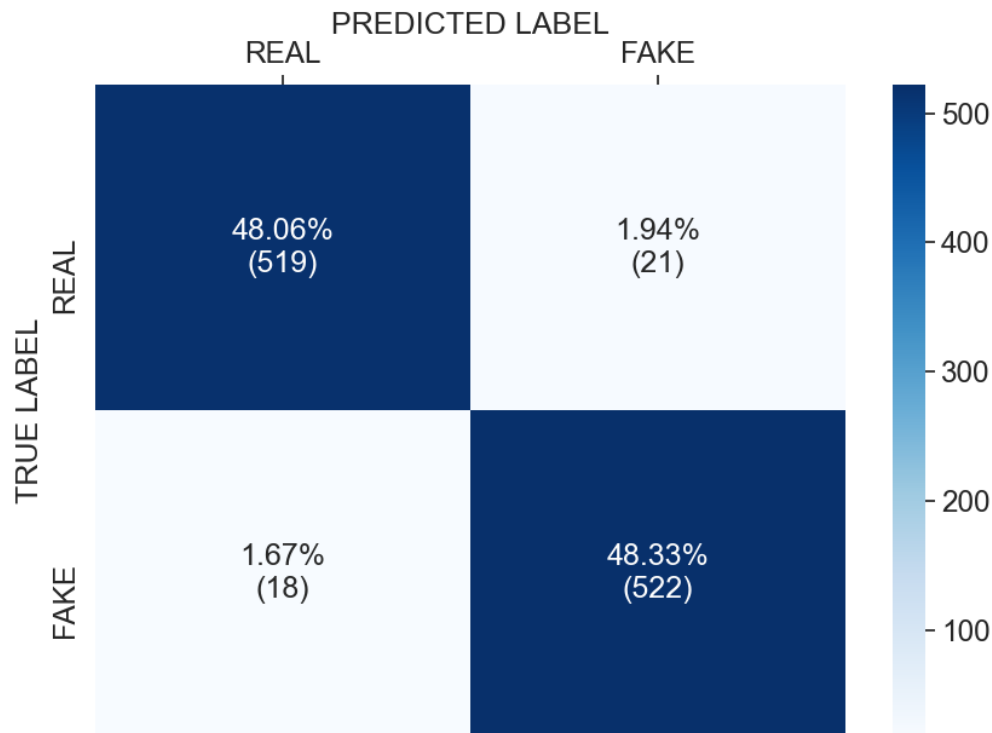


Figura (23) – Matriz de confusão da IDALina para dados fora da base de treinamento e validação.

A Figura 23 representa uma tabela conhecida em *machine learning*, chamada Matriz de Confusão. Ela mostra o desempenho do modelo por meio do cruzamento dos rótulos reais (notícias rotuladas previamente como falsas ou verdadeiras) com os rótulos classificados pela rede. Nota-se que o percentual de falsos positivos e falsos negativos é muito pequeno em relação à capacidade de acerto do modelo. Percebe-se também que, dos percentuais de erro, o percentual de falso positivo é o menor (1,67%), o que, para o problema de detecção de notícias falsas, possui uma maior significância (afinal, é mais grave validar uma notícia falsa do que o contrário).

Desta forma, percebe-se que o modelo se manteve próximo em relação às medidas apresentadas na base de validação. Era esperado que não fossem iguais, afinal os resultados obtidos da base de validação são fruto de sucessivos aprendizados de uma rede neural ao longo das épocas na base de treinamento.

6 Conclusão

Este trabalho propõe uma arquitetura de aprendizagem profunda por meio de camadas convolucionais e recorrentes, chamada de IDALina, que visa resolver o problema da proliferação de notícias falsas em português na Internet. Inicialmente, foi realizada uma revisão sistemática afim de identificar as oportunidades de pesquisa e auxiliar na construção do modelo. Esta etapa inicial foi fundamental para o conhecimento das técnicas e conceitos de redes neurais, aprendizagem de máquina, aprendizagem profunda e processamento de linguagem natural (NLP).

Uma vez construído o modelo, foram executados, inicialmente, experimentos de análise do *dataset*. Esta análise foi de suma importância para evitar possíveis gargalos no processamento de texto. Em seguida, realizou-se um experimento comparativo dos etiquetadores discutidos nesta dissertação, concluindo que o etiquetador de Brill foi o que apresentou resultados mais promissores.

Então este projeto chegou no momento mais complexo de análise preliminar, que foi os ajustes de hiperparâmetros. Para isso, o autor precisou se aprofundar nos conceitos matemáticos do Gradiente Descendente Estocástico (ver Anexo A.1) para entender o funcionamento de cada função de otimização. No final, concluiu-se que o otimizador de Adam com *learning rate* de 0.001 apresentou melhores resultados. Então, foi analisado, em seguida, o tamanho do lote. Ele basicamente influencia no cálculo do gradiente e, por isso, pode influenciar drasticamente na convergência ótima do modelo. O experimento mostrou que, o tamanho de lote que melhor resolve o problema desta dissertação é o tamanho de lote de 128.

Finalmente, nos experimentos que geraram os resultados finais desta dissertação, o modelo proposto (IDALina) foi comparado com várias arquiteturas distintas e trabalhos com soluções consideradas estado-da-arte, superando em todas as medidas de performance todos estes modelos. A IDALina, obteve uma acurácia de 98,1%, com uma taxa de falso positivo de 1,9%.

Desta forma, conclui-se que todos os objetivos gerais e específicos apresentados na Seção 1.1 foram atingidos por meio desta dissertação.

6.1 Trabalhos Futuros

Esta dissertação foi proposta com objetivos relativamente amplos pela sua natureza inovadora e, relativamente, inexplorada acerca do tema. Entretanto, ao longo do processo de amadurecimento técnico e acadêmico, o autor deparou-se com diversas técnicas promissoras

na classificação de texto mas que acabaram ficando fora da presente dissertação devido à limitações de tempo e recurso computacional.

Os experimentos realizados na IDALina não consideraram o uso de algoritmos de inicialização de pesos nos neurônios das camadas da arquitetura e seu impacto na rede. Embora o autor não tenha se aprofundado no assunto e, portanto, não teria condições de realizar inferências acerca de como estes algoritmos seriam aplicados, é válido destacar que estes algoritmos podem, inclusive, ser combinados entre as camadas da IDALina e seu impacto avaliado em um próximo estudo.

Uma característica observada pelo autor após a construção e experimentação do modelo reside na tendência em incorporar mecanismos de atenção em suas soluções (GUO, 2018; GUO et al., 2018; LIU et al., 2018; YU et al., 2019; GENG et al., 2019; B et al., 2018).

Outra contribuição importante para esta linha de pesquisa seria a construção de um *dataset* apropriado para estudos de detecção de notícias falsas em português. A proposta seria fazer um estudo similar ao de Monteiro, Santos e B (2018), com a contribuição de ser uma base de notícias mais atualizadas. Uma possibilidade seria até mesmo, incluir notícias de redes sociais.

A arquitetura proposta foi construída para tratar apenas características relacionadas ao texto. Uma proposta de trabalho futuro seria construir um classificador que detecta notícias falsas voltadas para redes sociais. Neste cenário, é preciso considerar, além de informação textual, imagens, engajamento social do usuário como características sociais a serem inseridas no modelo. Nesta proposta, a IDALina poderia cuidar do ramo textual com algumas possíveis adaptações (afinal, o conteúdo textual de notícias advindas de sites é diferente de notícias de redes sociais). Já as características de imagem e sociais seriam tratadas por outras estruturas, que, ao final, concatenariam a informação a ser classificada em uma arquitetura conhecida como multimodal. Alguns estudos encontrados pelo autor abordam propostas similares (JIN; CAO; GUO, 2017; WANG et al., 2018; MA et al., 2016a; SABIR et al., 2018).

6.2 Considerações Finais

Contudo, a principal contribuição deste trabalho constitui em fomentar a discussão científica e a pesquisa nacional no sentido de encontrar uma possível solução para o problema da disseminação de notícias falsas na Internet no cenário nacional. Sabe-se que o Brasil não é o único país afetado por esse problema quando comparado com os demais. Mas, em termos de pesquisa científica, é um dos que está mais distante de uma solução, haja vista a escassez de publicações nacionais acerca deste tema.

Em relação ao autor, o trabalho deu a oportunidade de aprender mais sobre aprendizagem de máquina e aprendizagem profunda, temas de pouco domínio prévio por parte do mesmo. Por não ter experiência com a pesquisa científica em geral, o autor também passou por uma curva de aprendizado que foi importante em seu processo de amadurecimento acadêmico ao longo do curso.

Por fim, espera-se que este estudo seja base de motivação para outras contribuições relacionadas ao tema no cenário nacional.

Referências

AHMED, H.; TRAORE, I.; SAAD, S. Detection of online fake news using n-gram analysis and machine learning techniques. *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, v. 10618, p. 127–138, 2017. Disponível em: <<http://link.springer.com/10.1007/978-3-319-69155-8>>. Citado na página 66.

AJAO, D. B. O.; ZARGARI, S. Sentiment aware fake news detection on online social networks. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, n. c, p. 2507–2511, 2019. Disponível em: <<http://arxiv.org/abs/1904.13355>>. Citado na página 66.

ALKHODAIR, S. A. et al. Detecting breaking news rumors of emerging topics in social media. *Information Processing and Management*, Elsevier, n. January, p. 1–13, 2019. ISSN 03064573. Disponível em: <<https://doi.org/10.1016/j.ipm.2019.02.016>>. Citado na página 66.

ALLCOTT, H.; GENTZKOW, M. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, v. 31, n. 2, p. 211–36, 2017. Citado 2 vezes nas páginas 29 e 61.

ALUÍSIO, S. et al. An account of the challenge of tagging a reference corpus for brazilian portuguese. In: SPRINGER. *International Workshop on Computational Processing of the Portuguese Language*. [S.l.], 2003. p. 110–117. Citado na página 37.

AMAZON. *Introduction to Amazon SageMaker Object2Vec*. 2020. Acessado em 21 de fevereiro de 2020. Disponível em: <<https://aws.amazon.com/pt/blogs/machine-learning/introduction-to-amazon-sagemaker-object2vec/>>. Citado 2 vezes nas páginas 15 e 45.

ARIF, A. et al. A closer look at the self-correcting crowd: Examining corrections in online rumors. In: ACM. *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. [S.l.], 2017. p. 155–168. Citado na página 32.

B, T. C. et al. Call attention to rumors : Deep attention based recurrent neural networks for early rumor detection. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. [S.l.]: Springer International Publishing, 2018. v. 1, p. 40–52. ISBN 9783030045036. Citado 3 vezes nas páginas 54, 66 e 94.

BAKSHY, E.; MESSING, S.; ADAMIC, L. A. Exposure to ideologically diverse news and opinion on facebook. *Science*, American Association for the Advancement of Science, v. 348, n. 6239, p. 1130–1132, 2015. Citado na página 31.

BANKO, M.; MOORE, R. C. Part-of-speech tagging in context. In: *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*. [S.l.: s.n.], 2004. p. 556–561. Citado na página 39.

BARBOSA, J. et al. Introdução ao processamento de linguagem natural usando python. *III Escola Regional de Informática do Piauí*, v. 1, p. 336–360, 2017. Citado na página 33.

- BARONI, M.; LENCI, A. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, MIT Press, v. 36, n. 4, p. 673–721, 2010. Citado na página 111.
- BASHEER, I. A.; HAJMEER, M. Artificial neural networks: fundamentals, computing, design, and application. *Journal of microbiological methods*, Elsevier, v. 43, n. 1, p. 3–31, 2000. Citado na página 51.
- BENGIO, Y. et al. A neural probabilistic language model. *Journal of machine learning research*, v. 3, n. Feb, p. 1137–1155, 2003. Citado na página 44.
- BENGIO, Y. et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, Now Publishers, Inc., v. 2, n. 1, p. 1–127, 2009. Citado na página 53.
- BHATTACHARJEE, S. D.; TALUKDER, A.; BALANTRAPU, B. V. Active learning based news veracity detection with feature weighting and deep-shallow fusion. *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017*, v. 2018-Janua, p. 556–565, 2018. Citado na página 66.
- BONDIELLI, A.; MARCELLONI, F. A survey on fake news and rumour detection techniques. *Information Sciences*, Elsevier, v. 497, p. 38–55, 2019. Citado 3 vezes nas páginas 15, 33 e 68.
- BONDIELLI, A.; MARCELLONI, F. A survey on fake news and rumour detection techniques. *Information Sciences*, Elsevier, v. 497, p. 38–55, 2019. Citado na página 25.
- BORDES, A.; CHOPRA, S.; WESTON, J. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*, 2014. Citado 2 vezes nas páginas 34 e 56.
- BORDIGNON, A. C. de A. et al. Natural language processing in business process identification and modeling: A systematic literature review. In: *Proceedings of the XIV Brazilian Symposium on Information Systems*. [S.l.: s.n.], 2018. p. 1–8. Citado na página 33.
- BRANCO, A.; SILVA, J. R. Evaluating solutions for the rapid development of state-of-the-art pos taggers for portuguese. In: *LREC*. [S.l.: s.n.], 2004. Citado na página 39.
- BRANTS, T. Tnt-a statistical part-of-speech tagger. *arXiv preprint cs/0003055*, 2000. Citado na página 39.
- BRILL, E. *A simple rule-based part of speech tagger*. [S.l.], 1992. Citado na página 36.
- BRILL, E. Some advances in transformation-based part of speech tagging. *arXiv preprint cmp-lg/9406010*, 1994. Citado 2 vezes nas páginas 15 e 36.
- CAI, G.; BI, M.; LIU, J. A novel rumor detection method based on labeled cascade propagation tree. *ICNC-FSKD 2017 - 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, IEEE, p. 2185–2194, 2018. Citado na página 66.
- CHEN, Y. et al. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected topics in applied earth observations and remote sensing*, IEEE, v. 7, n. 6, p. 2094–2107, 2014. Citado na página 54.

- 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. Citado 3 vezes nas páginas 43, 44 e 54.
- COLLOBERT, R. et al. Natural language processing (almost) from scratch. *Journal of machine learning research*, v. 12, n. Aug, p. 2493–2537, 2011. Citado 3 vezes nas páginas 37, 43 e 56.
- DEERWESTER, S. et al. Indexing by latent semantic analysis. *Journal of the American society for information science*, Wiley Online Library, v. 41, n. 6, p. 391–407, 1990. Citado na página 43.
- DENG, L.; LIU, Y. *Deep learning in natural language processing*. [S.l.]: Springer, 2018. Citado 2 vezes nas páginas 15 e 52.
- DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. Citado na página 66.
- DICKSON, P. *The Origins of Writerly Words*. 2014. <<https://time.com/82601/the-origins-of-writerly-words/>>. Citado na página 30.
- DUMAIS, S. T. et al. Using latent semantic analysis to improve access to textual information. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. [S.l.: s.n.], 1988. p. 281–285. Citado na página 43.
- EISENSTEIN, J. *Introduction to natural language processing*. [S.l.]: Mit Press, 2019. Citado na página 34.
- FILHO, P. B.; PARDO, T. A. S.; ALUÍSIO, S. An evaluation of the brazilian portuguese liwc dictionary for sentiment analysis. In: *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*. [S.l.: s.n.], 2013. Citado na página 51.
- FINGER, M. Técnicas de otimização da precisão empregadas no etiquetador tycho brahe. *Proceedings of PROPOR*, São Paulo, p. 141–154, 2000. Citado na página 37.
- FONSECA, E.; ROSA, J. L. G. Mac-morpho revisited: Towards robust part-of-speech tagging. In: *Proceedings of the 9th Brazilian symposium in information and human language technology*. [S.l.: s.n.], 2013. Citado na página 37.
- FONSECA, E. R.; ROSA, J. L. G.; ALUÍSIO, S. M. Evaluating word embeddings and a revised corpus for part-of-speech tagging in portuguese. *Journal of the Brazilian Computer Society*, Springer, v. 21, n. 1, p. 1–14, 2015. Citado na página 37.
- FOURNEY, A. et al. Geographic and temporal trends in fake news consumption during the 2016 us presidential election. In: ACM. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. [S.l.], 2017. p. 2071–2074. Citado na página 27.
- FREITAS, C. Sobre a construção de um léxico da afetividade para o processamento computacional do português. *Revista Brasileira de Linguística Aplicada*, SciELO Brasil, v. 13, n. 4, p. 1031–1059, 2013. Citado na página 50.

GANEGEDARA, T. *Natural Language Processing with TensorFlow: Teach language to machines using Python's deep learning library*. [S.l.]: Packt Publishing Ltd, 2018. Citado na página 49.

GENG, Y. et al. Rumor detection on social media: A multi-view model using self-attention mechanism. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2019. v. 11538, p. 339–352. ISBN 978-3-030-22743-2. Disponível em: <<http://link.springer.com/10.1007/978-3-030-22744-9>>. Citado 2 vezes nas páginas 66 e 94.

GIUDICE, K. D. Crowdsourcing credibility: The impact of audience feedback on web page credibility. In: AMERICAN SOCIETY FOR INFORMATION SCIENCE. *Proceedings of the 73rd ASIS&T Annual Meeting on Navigating Streams in an Information Ecosystem-Volume 47*. [S.l.], 2010. p. 59. Citado na página 32.

GLOROT, X.; BORDES, A.; BENGIO, Y. Domain adaptation for large-scale sentiment classification: A deep learning approach. In: *Proceedings of the 28th international conference on machine learning (ICML-11)*. [S.l.: s.n.], 2011. p. 513–520. Citado na página 56.

GOLDBERG, Y. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, Morgan & Claypool Publishers, v. 10, n. 1, p. 1–309, 2017. Citado 5 vezes nas páginas 43, 44, 51, 53 e 81.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. Deep learning (adaptive computation and machine learning series). *DOI*, v. 10, p. 1762–1766, 2016. Citado 2 vezes nas páginas 83 e 85.

GUO, H. et al. Rumor detection with hierarchical social attention network. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, p. 943–951, 2018. Citado 2 vezes nas páginas 66 e 94.

GUO, Z. Rumor detection on twitter with hierarchical attention neural networks. *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, IEEE, p. 783–787, 2018. Citado na página 94.

HALÁCSY, P.; KORNAI, A.; ORAVECZ, C. Hunpos-an open source trigram tagger. 2007. Citado na página 39.

HARRIS, Z. S. Distributional structure. *Word*, Taylor & Francis, v. 10, n. 2-3, p. 146–162, 1954. Citado na página 43.

HAYKIN, S.; NETWORK, N. Neural networks: A comprehensive foundation. *Neural networks*, v. 2, n. 2004, p. 41, 2004. Citado 2 vezes nas páginas 15 e 57.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Citado 3 vezes nas páginas 15, 57 e 58.

HOTH, A.; NÜRNBERGER, A.; PAASS, G. A brief survey of text mining. In: CITESEER. *Ldv Forum*. [S.l.], 2005. v. 20, n. 1, p. 19–62. Citado na página 34.

- HUANG, X.; LIN, J.; DEMNER-FUSHMAN, D. Evaluation of pico as a knowledge representation for clinical questions. In: AMERICAN MEDICAL INFORMATICS ASSOCIATION. *AMIA annual symposium proceedings*. [S.l.], 2006. v. 2006, p. 359. Citado na página 59.
- INDURKHYA, N.; DAMERAU, F. J. *Handbook of natural language processing*. [S.l.]: Chapman and Hall/CRC, 2010. Citado na página 33.
- JAMIESON, K. H.; CAPPELLA, J. N. *Echo chamber: Rush Limbaugh and the conservative media establishment*. [S.l.]: Oxford University Press, 2008. Citado na página 25.
- JEAN, S. et al. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*, 2014. Citado na página 56.
- JIN, Z.; CAO, J.; GUO, H. Multimodal fusion with recurrent neural networks for rumor detection on microblogs. *Proceedings of the 25th ACM international conference on Multimedia (MM' 17)*, p. 795–803, 2017. Citado 3 vezes nas páginas 54, 66 e 94.
- JIN, Z. et al. News verification by exploiting conflicting social viewpoints in microblogs. In: *Thirtieth AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2016. Citado na página 32.
- JOHNSON, R.; ZHANG, T. Semi-supervised convolutional neural networks for text categorization via region embedding. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2015. p. 919–927. Citado 2 vezes nas páginas 33 e 54.
- KAWAKAMI, K. Supervised sequence labelling with recurrent neural networks. *Ph. D. dissertation, PhD thesis. Ph. D. thesis*, Technical University of Munich, 2008. Citado na página 53.
- KESKAR, N. S. et al. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016. Citado na página 86.
- KHANDELWAL, S.; LECOUTEUX, B.; BESACIER, L. *Comparing GRU and LSTM for automatic speech recognition*. Tese (Doutorado) — LIG, 2016. Citado na página 88.
- KIM, J. et al. Leveraging the crowd to detect and reduce the spread of fake news and misinformation. In: ACM. *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. [S.l.], 2018. p. 324–332. Citado na página 32.
- KINOSHITA, J.; SALVADOR, L. do N.; MENEZES, C. E. D. de. Cogroo: a brazilian-portuguese grammar checker based on the cetenfolha corpus. In: CITESEER. *LREC*. [S.l.], 2006. p. 2190–2193. Citado na página 37.
- KINOSHITA, J. et al. Cogroo-an openoffice grammar checker. In: IEEE. *Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007)*. [S.l.], 2007. p. 525–530. Citado na página 37.
- KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. Citeseer, 2007. Citado 2 vezes nas páginas 59 e 63.

KITCHENHAM, B. A.; BUDGEN, D.; BRERETON, O. P. Using mapping studies as the basis for further research—a participant-observer case study. *Information and Software Technology*, Elsevier, v. 53, n. 6, p. 638–651, 2011. Citado na página 59.

KUMAR, A.; SANGWAN, S. R.; NAYYAR, A. Rumour veracity detection on twitter using particle swarm optimized shallow classifiers. *Multimedia Tools and Applications*, Multimedia Tools and Applications, 2019. Citado na página 66.

LECUN, Y.; BENGIO, Y. et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, v. 3361, n. 10, p. 1995, 1995. Citado na página 54.

LI, L.; CAI, G.; CHEN, N. A rumor events detection method based on deep bidirectional gru neural network. *2018 3rd IEEE International Conference on Image, Vision and Computing, ICIVC 2018*, IEEE, p. 755–759, 2018. Citado 2 vezes nas páginas 54 e 66.

LIN, D. et al. An information-theoretic definition of similarity. In: *Icml*. [S.l.: s.n.], 1998. v. 98, n. 1998, p. 296–304. Citado na página 113.

LIU, Q. et al. Mining significant microblogs for misinformation identification : An attention-based approach. *ACM Transactions on Intelligent Systems and Technology*, v. 9, n. 5, 2018. Citado 2 vezes nas páginas 66 e 94.

LIU, Y.; JIN, X.; SHEN, H. Towards early identification of online rumors based on long short-term memory networks. *Information Processing and Management*, Elsevier, v. 56, n. 4, p. 1457–1467, 2019. ISSN 03064573. Disponível em: <<https://doi.org/10.1016/j.ipm.2018.11.003>>. Citado na página 66.

LIU, Y.; WU, Y.-f. B. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. *Thirty-Second AAAI Conference on Artificial Intelligence*, p. 354–361, 2018. Citado 2 vezes nas páginas 25 e 66.

MA, J. et al. Detecting rumors from microblogs with recurrent neural networks. In: *IJCAI*. [S.l.: s.n.], 2016. p. 3818–3824. Citado 2 vezes nas páginas 54 e 94.

MA, J. et al. Detect rumors using time series of social context information on microblogging websites. *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management - CIKM '15*, n. October, p. 1751–1754, 2016. Disponível em: <<http://dl.acm.org/citation.cfm?doid=2806416.2806607>>. Citado na página 66.

MA, J.; GAO, W.; WONG, K.-F. Detect rumors in microblog posts using propagation structure via kernel learning. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 2017. p. 708–717. Citado na página 66.

MA, J.; GAO, W.; WONG, K.-F. Rumor detection on twitter with tree-structured recursive neural networks. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, p. 1980–1989, 2018. Citado 2 vezes nas páginas 54 e 66.

MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. *Introduction to information retrieval*. [S.l.]: Cambridge university press, 2008. Citado na página 33.

- MÀRQUEZ, L. et al. *Semantic role labeling: an introduction to the special issue*. [S.l.]: MIT Press, 2008. Citado na página 37.
- MARTINEZ-GIL, J. An overview of textual semantic similarity measures based on web intelligence. *Artificial Intelligence Review*, Springer, v. 42, n. 4, p. 935–943, 2014. Citado na página 113.
- MEDEIROS, F. D.; BRAGA, R. B. Fake news detection in social media: A systematic review. In: *XVI Brazilian Symposium on Information Systems*. [s.n.], 2020. (SBSI'20). Disponível em: <<https://doi.org/10.1145/3411564.3411648>>. Citado na página 26.
- MENEZES, C. E. D. d.; NETO, J. J. Um método híbrido para a construção de etiquetadores morfológicos, aplicado à língua portuguesa, baseado em autômatos adaptativos. In: *Proceedings of the 2nd Conferencia Iberoamericana en Sistemas, Cibernética e Informática (CISCI'2002), Orlando, USA*. [S.l.: s.n.], 2002. Citado na página 37.
- MIKOLOV, T. et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. Citado 3 vezes nas páginas 43, 44 e 45.
- MIKOLOV, T.; LE, Q. V.; SUTSKEVER, I. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013. Citado 2 vezes nas páginas 44 e 45.
- MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2013. p. 3111–3119. Citado 4 vezes nas páginas 44, 45, 49 e 81.
- MITKOV, R. *The Oxford handbook of computational linguistics*. [S.l.]: Oxford University Press, 2004. Citado na página 34.
- MONTEIRO, R. A.; SANTOS, R. L. S.; B, T. A. S. P. Contributions to the study of fake news in portuguese: New corpus and automatic detection results. In: *International Conference on Computational Processing of the Portuguese Language*. [S.l.]: Springer International Publishing, 2018. p. 324–334. Citado 11 vezes nas páginas 17, 26, 65, 66, 67, 78, 87, 89, 90, 91 e 94.
- NICKERSON, R. Confirmation bias: A ubiquitous phenomenon in many guises. v. 2, p. 175–220, 06 1998. Citado na página 31.
- NLTK. *Natural Language Toolkit*. 2020. Acessado em 8 de janeiro de 2020. Disponível em: <<http://www.gnu.org/philosophy/free-sw.pt.html>>. Citado na página 78.
- NYHAN, B.; REIFLER, J. When corrections fail: The persistence of political misperceptions. *Political Behavior*, Springer, v. 32, n. 2, p. 303–330, 2010. Citado na página 31.
- OSHIKAWA, R.; QIAN, J.; WANG, W. Y. A survey on natural language processing for fake news detection. 2018. Citado 2 vezes nas páginas 25 e 56.
- OSHIKAWA, R.; QIAN, J.; WANG, W. Y. A survey on natural language processing for fake news detection. *arXiv preprint arXiv:1811.00770*, 2018. Citado na página 32.

- PENNEBAKER, J. W.; FRANCIS, M. E.; BOOTH, R. J. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, v. 71, n. 2001, p. 2001, 2001. Citado na página 50.
- PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. [S.l.: s.n.], 2014. p. 1532–1543. Citado 4 vezes nas páginas 34, 44, 49 e 50.
- PETERSEN, K. et al. Systematic mapping studies in software engineering. In: *Ease*. [S.l.: s.n.], 2008. v. 8, p. 68–77. Citado na página 59.
- POGUE, D. How to stamp out fake news. *Scientific American*, Scientific American, v. 316, n. 2, p. 24–24, 2017. Citado na página 25.
- POTTHAST, M. et al. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*, 2017. Citado na página 30.
- QIU, X. et al. Ensemble deep learning for regression and time series forecasting. In: *IEEE. 2014 IEEE symposium on computational intelligence in ensemble learning (CIEL)*. [S.l.], 2014. p. 1–6. Citado na página 54.
- RATNAPARKHI, A. A maximum entropy model for part-of-speech tagging. In: *Conference on empirical methods in natural language processing*. [S.l.: s.n.], 1996. Citado na página 39.
- REIS, J. C. et al. Supervised learning for fake news detection. *IEEE Intelligent Systems*, v. 34, n. 2, p. 76–81, 2019. ISSN 19411294. Citado na página 66.
- RESNIK, P. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*, 1995. Citado 3 vezes nas páginas 15, 113 e 114.
- RONG, X. Word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014. Citado 3 vezes nas páginas 15, 46 e 48.
- RUCHANSKY, N. Csi : A hybrid deep model for fake news detection. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, p. 797–806, 2017. Citado 3 vezes nas páginas 25, 32 e 66.
- SABIR, E. et al. Deep multimodal image-repurposing detection. *2018 ACM Multimedia Conference (MM '18)*, v. 2, p. 1337–1345, 2018. Citado na página 94.
- SANTOS, C. N. D.; MILIDIÚ, R. L.; RENTERÍA, R. P. Portuguese part-of-speech tagging using entropy guided transformation learning. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2008. p. 143–152. Citado na página 37.
- SETHI, R. J. Crowdsourcing the verification of fake news and alternative facts. In: *ACM. Proceedings of the 28th ACM Conference on Hypertext and Social Media*. [S.l.], 2017. p. 315–316. Citado na página 32.
- SHU, K.; MAHUDESWARAN, D.; LIU, H. Fakenewstracker: a tool for fake news collection, detection, and visualization. *Computational and Mathematical Organization Theory*, Springer US, n. 0123456789, 2018. ISSN 1572-9346. Disponível em: <<https://doi.org/10.1007/s10588-018-09280-3>>. Citado 2 vezes nas páginas 65 e 66.

- SHU, K.; WANG, S. Beyond news contents : The role of social context for fake news detection. *The Twelfth ACM International Conference on Web Search and Data Mining (WSDM '19)*, n. i, p. 312–320, 2019. Citado na página 66.
- SILVA, R. M. et al. Towards automatically filtering fake news in portuguese. *Expert Systems with Applications*, Elsevier, p. 113199, 2020. Citado 8 vezes nas páginas 26, 65, 66, 67, 87, 89, 90 e 91.
- SINGH, J. P.; RANA, N. P.; DWIVEDI, Y. K. Rumour veracity estimation with deep learning for twitter. In: *International Working Conference on Transfer and Diffusion of IT*. Springer International Publishing, 2019. v. 558, p. 351–363. ISBN 978-3-030-20670-3. Disponível em: <<http://link.springer.com/10.1007/978-3-030-20671-0>>. Citado na página 66.
- SINGHANIA, N. F. S.; RAO, S. 3han: A deep neural network for fake news detection. *Journal of the Society of Mechanical Engineers*, v. 90, n. 823, p. 758–759, 2017. Citado na página 54.
- STEINMACHER, I.; CHAVES, A. P.; GEROSA, M. A. Awareness support in distributed software development: A systematic review and mapping of the literature. *Computer Supported Cooperative Work (CSCW)*, Springer, v. 22, n. 2-3, p. 113–158, 2013. Citado na página 63.
- SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2014. p. 3104–3112. Citado na página 56.
- TACCHINI, E. et al. Some like it hoax: Automated fake news detection in social networks. *arXiv preprint arXiv:1704.07506*, 2017. Citado na página 32.
- TANG, D. et al. Learning sentiment-specific word embedding for twitter sentiment classification. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. [S.l.: s.n.], 2014. p. 1555–1565. Citado na página 34.
- TAUSCZIK, Y. R.; PENNEBAKER, J. W. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of language and social psychology*, Sage Publications Sage CA: Los Angeles, CA, v. 29, n. 1, p. 24–54, 2010. Citado na página 50.
- THANAKI, J. *Python natural language processing*. [S.l.]: Packt Publishing Ltd, 2017. Citado 2 vezes nas páginas 35 e 81.
- TSCHIATSCHEK, S. et al. Detecting fake news in social networks via crowdsourcing. *arXiv preprint arXiv:1711.09025*, 2017. Citado na página 32.
- TUFEKCI, Z. Facebook said its algorithms do help form echo chambers, and the tech press missed it. *New Perspectives Quarterly*, v. 32, n. 3, p. 9–12, 2015. Citado 2 vezes nas páginas 25 e 31.
- TURNEY, P. D. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In: SPRINGER. *European conference on machine learning*. [S.l.], 2001. p. 491–502. Citado na página 111.

- VARGO, C. J.; GUO, L.; AMAZEEN, M. A. The agenda-setting power of fake news: A big data analysis of the online media landscape from 2014 to 2016. *New Media & Society*, SAGE Publications Sage UK: London, England, v. 20, n. 5, p. 2028–2049, 2018. Citado na página 32.
- VEDOVA, M. L. D. et al. Automatic online fake news detection combining content and social signals. *2018 22nd Conference of Open Innovations Association (FRUCT)*, v. 79, n. 4, p. 273–279, 2018. ISSN 2305-7254. Citado na página 66.
- VICARIO, M. D. et al. Polarization and fake news: Early warning of potential misinformation targets. *ACM Transactions on the Web (TWEB)*, v. 13, n. 2, 2018. Disponível em: <<http://arxiv.org/abs/1802.01400>>. Citado na página 66.
- WANG, Y. et al. Eann : Event adversarial neural networks for multi-modal fake news detection. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, p. 849–857, 2018. Citado 2 vezes nas páginas 66 e 94.
- WARD L ROSS, E. R. E. T. A.; BROWN, T. Naive realism in everyday life: Implications for social conflict and misunderstanding. In: *Values and knowledge*. [S.l.]: Psychology Press, 1997. p. 103–135. Citado na página 31.
- WARDLE, C. *Fake news. It's complicated*. 2017. <<https://firstdraftnews.org/fake-news-complicated/>>. Citado na página 29.
- WU, L.; LIU, H. Tracing fake-news footprints : Characterizing social media messages by how they propagate. In: *Proceedings of WSDM 2018: The Eleventh ACM International Conference on Web Search and Data Mining*. [S.l.: s.n.], 2018. p. 637–645. ISBN 9781450355810. Citado na página 66.
- XU, N.; CHEN, G.; MAO, W. Mnrd: A merged neural model for rumor detection in social media. *Proceedings of the International Joint Conference on Neural Networks, IEEE*, v. 2018-July, p. 1–7, 2018. Citado na página 66.
- YANG, Z. et al. Hierarchical attention networks for document classification. In: *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*. [S.l.: s.n.], 2016. p. 1480–1489. Citado na página 54.
- YIN, W. et al. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*, 2017. Citado na página 88.
- YU, F. et al. A convolutional approach for misinformation identification. *IJCAI International Joint Conference on Artificial Intelligence*, p. 3901–3907, 2017. ISSN 10450823. Citado na página 66.
- YU, F. et al. Attention-based convolutional approach for misinformation identification from massive and noisy microblog posts. *Computers and Security*, Elsevier Ltd, v. 83, p. 106–121, 2019. ISSN 01674048. Disponível em: <<https://doi.org/10.1016/j.cose.2019.02.003>>. Citado 2 vezes nas páginas 66 e 94.
- ZHAI, C.; MASSUNG, S. *Text data management and analysis: a practical introduction to information retrieval and text mining*. [S.l.]: Association for Computing Machinery and Morgan & Claypool, 2016. Citado 2 vezes nas páginas 15 e 42.

- ZHANG, Q. et al. Reply-aided detection of misinformation via bayesian deep learning. *The World Wide Web Conference (WWW' 19)*, n. February, p. 2333–2343, 2019. Citado na página 66.
- ZHANG, X. et al. Deep neural network hyperparameter optimization with orthogonal array tuning. In: SPRINGER. *International Conference on Neural Information Processing*. [S.l.], 2019. p. 287–295. Citado na página 80.
- ZHANG, X.; GHORBANI, A. A. An overview of online fake news: Characterization, detection, and discussion. *Information Processing & Management*, Elsevier, 2019. Citado na página 56.
- ZHANG, Y.; WALLACE, B. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*, 2015. Citado na página 87.
- ZHOU, Z. et al. A c-gru neural network for rumors detection. *Proceedings of 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems, CCIS 2018*, IEEE, p. 704–708, 2019. Citado na página 66.
- ZUBIAGA, A.; LIAKATA, M.; PROCTER, R. Exploiting context for rumour detection in social media. In: SPRINGER. *International Conference on Social Informatics*. [S.l.], 2017. p. 109–123. Citado na página 66.
- ZULQARNAIN, M. et al. A comparative review on deep learning models for text classification. *Indones. J. Electr. Eng. Comput. Sci*, v. 19, n. 1, p. 325–335, 2020. Citado na página 88.

Apêndices

APÊNDICE A – Medidas de Associação

Para identificar quais contextos são relevantes e quais não são em uma representação distribuída, são necessárias algumas medidas de associação entre palavras e contextos. Elas irão auxiliar os modelos na geração dos vetores de palavras.

A.1 Matriz Palavra-Contexto

Em NLP alguns trabalhos obtêm as características de uma palavra utilizando uma matriz $\mathbf{M}_{[i,j]}$ (TURNERY, 2001; BARONI; LENCI, 2010), onde cada linha i representa uma palavra e cada coluna j representa um contexto.

Mais formalmente, considerando V_W como o conjunto de palavras (conhecido como dicionário) e V_C o conjunto de todos os contextos possíveis, podemos assumir que cada palavra e cada contexto podem ser mapeados através de uma matriz. Desta forma, sendo w_i a i -ésima palavra no vocabulário e c_j os contextos da j -ésima palavra, temos que a matriz $\mathbf{M}_{[i,j]} \in \mathbb{R}^{|V_W| \times |V_C|}$ é a matriz palavra-contexto, definida por $\mathbf{M}_{[i,j]}^f = f(w_i, c_j)$, onde f é uma medida de associação entre uma palavra e um contexto.

A.1.0.1 Pointwise Mutual Information (PMI)

Como o próprio nome diz, abordagens de hipóteses distribuídas baseadas em contagem, normalmente, associam sua função f a um número baseado em contagem de palavras num contexto. Ou seja, considerando $count(w, c)$ como o número de vezes em que a palavra w ocorreu no contexto c no corpus D , f poderia ser

$$f(w, c) = p(w, c) = \frac{count(w, c)}{|D|}. \quad (\text{A.1})$$

Entretanto, isso pode gerar um efeito indesejado ao atribuir um peso muito grande para relações palavra-contexto que aparecem em contextos muito comuns. Por exemplo, considere o contexto de uma palavra ser a palavra antecessora de outra. Então para uma palavra como *cão*, as saídas *o cão* e *um cão* receberão um peso maior que *pequeno cão* ou *belo cão*, mesmo que estes últimos carreguem um significado maior.

Para conter este efeito, usa-se uma métrica chamada Informação Mútua Ponto-a-Ponto (ou *Pointwise Mutual Information - PMI*). Ela calcula a razão entre a probabilidade

conjunta e suas probabilidades marginais.

$$\text{PMI}(w, c) = \log \frac{p(w, c)}{p(w)p(c)} \quad (\text{A.2})$$

Entretanto, o PMI possui a desvantagem de atribuir altos valores a eventos raros. Imagine uma palavra e um contexto ocorram apenas 1 vez (e ocorrem simultaneamente). Assim, nenhum termo anularia a razão contida no logaritmando da Equação A.2. No entanto, como o uso de *word embeddings* considera que existe um corpus D grande o suficiente para estruturar corretamente a linguagem a ser adotada, logo, pela Equação A.1, teríamos

$$\text{PMI}(w, c) = \log \frac{\frac{\text{count}(w, c)}{|D|}}{\frac{\text{count}(w)}{|D|} \frac{\text{count}(c)}{|D|}} = \log \frac{\overset{\rightarrow 1}{\text{count}(w, c)} \cdot |D|}{\overset{\rightarrow 1}{\text{count}(w)} \cdot \overset{\rightarrow 1}{\text{count}(c)}} = \log |D| \quad (\text{A.3})$$

Desta forma, o valor atribuído pela métrica PMI (ver Apêndice A.1.0.1) a para este evento raro seria muito alta. Portanto, é aconselhável definir algum valor mínimo de co-ocorrências onde se pode considerar a contagem e utilizar a métrica PMI ou simplesmente desconsiderar eventos raros.

APÊNDICE B – Medidas de Similaridade

Medir semanticamente a similaridade de uma palavra, frase, parágrafo ou texto é um problema fundamental para várias aplicações computacionais (MARTINEZ-GIL, 2014). A principal abordagem aplicada a este problema utiliza uma topologia baseada em nós¹ (*node-based*), que carrega o conceito de Conteúdo da Informação (*informational content* - *IC*). Ela compara as propriedades entre dois termos, que podem ser relativas aos próprios termos, aos seus ancestrais ou aos seus descendentes.

B.1 Informational Content

O IC de um termo é uma propriedade comumente utilizada, e diz o quão específico e informativo um termo é, e pode ser calculado com a fórmula:

$$IC = -\log p(c) \quad (\text{B.1})$$

Onde $p(c)$ é a probabilidade de ocorrência de c em um corpus específico, geralmente estimado pela sua frequência de anotação. Intuitivamente, pode-se notar que quanto mais abstrato for o termo, menor é a informação que ele carrega em si.

B.2 Similaridade de Resnik

Resnik (1995) então definiu a similaridade entre dois conceitos como

$$\text{sim}_{RES}(c_1, c_2) = \max_{c \in S(c_1, c_2)} IC(c), \quad (\text{B.2})$$

onde $S(c_1, c_2)$ consiste no conjunto de todos os ancestrais comuns entre os conceitos c_1 e c_2 .

B.3 Similaridade de Lin

Lin et al. (1998) se baseou no trabalho de (RESNIK, 1995) para relacionar o conteúdo da informação do ancestral comum mais próximo com o termo sendo comparado, resultando na seguinte medida

$$\text{sim}_{LIN}(c_1, c_2) = \frac{2 \cdot IC(LCS(c_1, c_2))}{IC(c_1) + IC(c_2)}. \quad (\text{B.3})$$

¹ Existem outras topologias, mas elas não serão o foco deste trabalho.

O menor ancestral comum (Lowest Common Subsumer - LCS) é o que possui, dos conceitos comparados a menor distância. A Figura 24 ilustra o conceito de menor ancestral comum, onde o LCS entre *nickel* e *dime* é *coin*; já o LCS entre *credit card* e *nickel* é o termo *medium of exchange*.

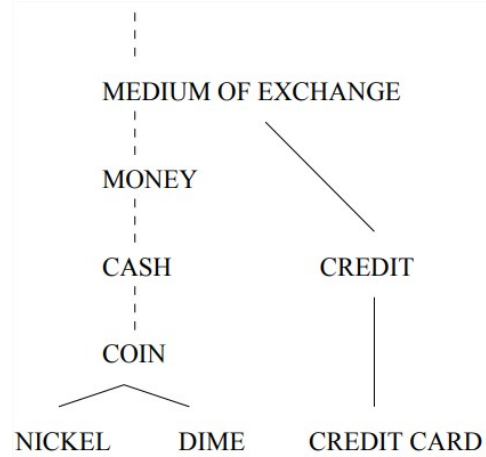


Figura (24) – Fragmento da taxonomia WordNet. Linhas sólidas representam ligações "é um" e linhas tracejadas indicam que nós foram omitidos. (RESNIK, 1995)

B.4 Cosseno

Uma medição comum, porém muito efetiva, é a medição pelo ângulo formado entre dois vetores

$$\text{sim}_{\cos}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} = \frac{\sum_i \mathbf{a}_i \cdot \mathbf{b}_i}{\sqrt{\sum_i (\mathbf{a}_i)^2} \sqrt{\sum_i (\mathbf{b}_i)^2}} \quad (\text{B.4})$$

Anexos

ANEXO A – Técnicas Relacionados a *Deep Learning*

A.1 Gradiente Descendente Estocástico

O algoritmo Gradiente Descendente Estocástico (do inglês *Stochastic Gradient Descent*, ou abreviadamente, SGD), se baseia em calcular repetidamente uma perda estimada sobre a base de treinamento através do cálculo dos gradientes das funções de perda estimada. Em outras palavras, é uma forma de encontrar os mínimos locais de uma determinada função objetivo.

Seja $f(x)$ a função objetivo que se quer minimizar. O gradiente descendente inicia com uma estimativa de solução x_0 , ou seja, um “chute” para o mínimo local. Para descobrir qual é a melhor direção para otimizar $f(x)$, calcula-se o gradiente nesse ponto. Geometricamente, o gradiente informará a inclinação da função em x_0 e a sua direção irá apontar para onde a função cresce mais rapidamente. Naturalmente, o próximo valor de x considera o oposto do gradiente. Logo, o gradiente entra no cálculo do próximo mínimo como um valor negativo. O algoritmo irá, eventualmente, convergir quando o gradiente for zero (ou aproximadamente zero), o que corresponde a um mínimo local. A equação abaixo representa a atualização de x durante a iteração t :

$$x_t = x_{t-1} - \eta \frac{\partial}{\partial x} f(x), \quad (\text{A.1})$$

onde η é a taxa de aprendizagem do gradiente descendente, que corresponde ao tamanho do passo na direção gradiente.

A.1.1 Batch Normalization

É consensual a importância de normalizar os dados de entrada antes de executar um modelo de predição. Entretanto, para redes neurais profundas, à medida que a informação é processada e enviada às camadas mais internas, ocorre um problema conhecido como mudança de covariável interna, do inglês *internal covariate shift*.

Por exemplo, imagine uma rede multi-CNN encadeada para resolver um problema qualquer de processamento de imagens. Considere que a primeira camada, para um certo pixel de uma imagem, aumenta o peso dela em 10 vezes; por sua vez, a camada seguinte reduz o peso em 10. Consequentemente, as duas camadas iriam se anular neste ponto, fornecendo informação não útil para o modelo.

Para resolver este problema, recorre-se à uma técnica conhecida como normalização em lote, do inglês *Batch Normalization*, que consiste em dividir o conjunto de treinamento em pequenos lotes, chamado *mini-batch*, normalizar os dados de saída da camada anterior. No final, a camada de normalização envia como saída a média e variância global dos valores recebidos.

A.1.2 Dropout

A modelagem de redes profundas propicia a resolução de problemas mais complexos, devido a sua capacidade de aprendizado provida pelas diversas camadas da rede. Por outro lado, por conta dessa diversidade de camadas e parâmetros, estas redes sofrem de um efeito conhecido como *overfitting*. O *overfitting* é uma condição em que se percebe que o modelo possui um acerto excelente na base de treinamento mas perdeu a capacidade de generalização e possui rendimento fraco na base de testes. Afim de evitar este problema, diversas técnicas foram propostas; elas são classificadas como Técnicas de Regularização.

Uma das principais técnicas de regularização é o Desligamento, ou *Dropout*. O Dropout é uma técnica em que neurônios selecionados aleatoriamente são ignorados durante o treinamento. Isso significa que sua contribuição para a função de ativação é temporariamente removida e quaisquer atualizações de peso não são aplicadas ao neurônio na época seguinte. Este processo é repetido a cada época. A Figura 25 ilustra o funcionamento da técnica.

A técnica de *dropout* pode ser entendida também como uma forma de acréscimo de dados (*data augmentation*). Afinal, a rede passa a se influenciar por um novo padrão de informação similar ao dado real.

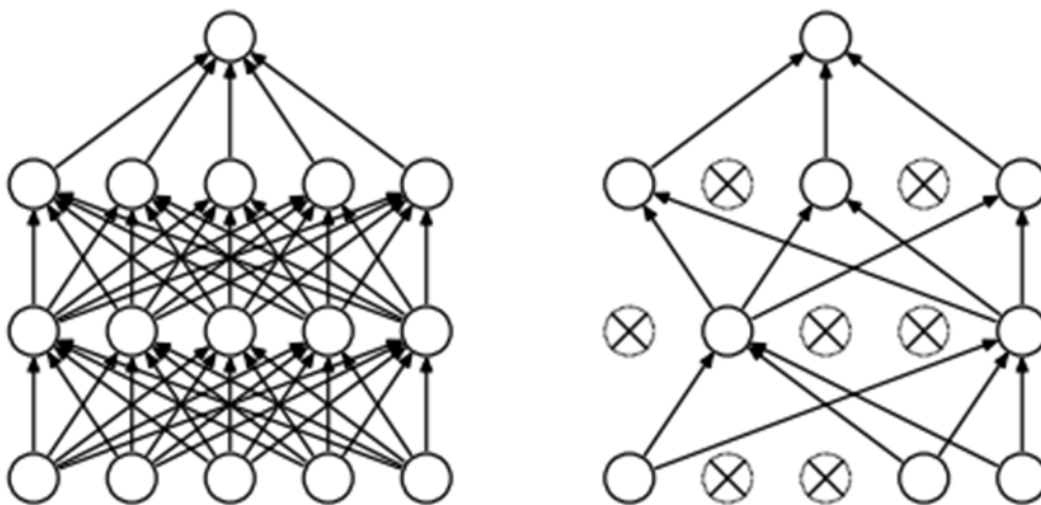


Figura (25) – Técnica de *dropout*. À rede completamente conectada da esquerda é submetida a um *dropout*, resultando na rede da figura à direita.