Universidade Estadual de Londrina

FABIANO SHIITI MARUMO

DEEP LEARNING PARA CLASSIFICAÇÃO DE FAKE NEWS POR SUMARIZAÇÃO DE TEXTO

LONDRINA
2018

FABIANO SHIITI MARUMO

DEEP LEARNING PARA CLASSIFICAÇÃO DE FAKE NEWS POR SUMARIZAÇÃO DE TEXTO

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Sylvio Barbon Junior

LONDRINA

2018

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

Marumo, Fabiano Shiiti.
Deep Learning para classificação de Fake News por sumarização de texto / Fabiano Shiiti Marumo. - Londrina, 2019.
52 f.

Orientador: Prof. Dr. Sylvio Barbon Jr. .
Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Graduação em Ciência da Computação, 2019.
Inclui bibliografia.

1. Aprendizado Profundo - TCC. 2. Notícias Falsas - TCC. 3. Mineração de Texto - TCC. I. , Prof. Dr. Sylvio Barbon Jr.. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Graduação em Ciência da Computação. III. Título.

FABIANO SHIITI MARUMO

DEEP LEARNING PARA CLASSIFICAÇÃO DE FAKE NEWS POR SUMARIZAÇÃO DE TEXTO

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA

Orientador: Prof. Dr. Sylvio Barbon Junior
Universidade Estadual de Londrina

Prof. Dr. Rafael Gomes Mantovani
Universidade Estadual de Londrina – UEL

Prof. Dr. Luiz Fernando Carvalho
Universidade Estadual de Londrina – UEL

Londrina, 20 de dezembro de 2018.

Este trabalho é uma dedicatória às pessoas que me apoiaram direta e indiretamente ao decorrer do ensino superior e uma forma de incentivo para mostrar que qualquer pessoa pode realizar seus sonhos, desde que tenha humildade, honestidade e gratidão.

AGRADECIMENTOS

Existem muitas pessoas ao qual devo meus sinceros agradecimentos, porém vou focar apenas a aqueles que participaram da minha graduação.

Uma das pessoas a qual devo meu profundos agradecimentos é o meu orientador Sylvio, que me propôs um tema para TCC que me despertou interesse e me motivou a desenvolvê-lo, me ajudou como inúmeras orientações e que não desistiu de mim.

Agradeço também à minha família, por ter sempre me proporcionado uma vida na qual não tenho reclamações, que apesar de vários problemas, me possibilitaram os meus estudos sem maiores preocupações.

Agradeço também aos meus amigos, que foram uma grande fonte de aprendizado e inspirações aos longos dos últimos anos, permitindo o meu desenvolvimento como pessoa e que me ajudaram durante os vários obstáculos que existiram.

Agradeço aos professores, que além de me ensinarem, foram muito compreensivos e pacientes com nós alunos.

A mim, por chegar até aqui, mas sem saber como foi possível.

E por último, mas também não menos importante, a Deus, que me permitiu chegar até aqui, sempre me fortalecendo e guiando o meu caminho.

"A beleza é a única coisa preciosa na vida.
É difícil encontrá-la - mas quem consegue
descobre tudo.
(Charles Chaplin)

MARUMO, FABIANO SHIITI. **Deep Learning para classificação de Fake News por sumarização de texto**. 2018. 54f. – Universidade Estadual de Londrina, Londrina, 2018.

RESUMO

Notícias sempre foram importantes para a sociedade, pois através delas informações são entregues às pessoas, seja por meio físico ou digital. Em uma sociedade em que o cidadão tem direito de acesso à informação, a propagação de notícias falsas vem se tornando uma preocupação. Conhecido por *Fake News*, o uso mal intencionado de informações pode comprometer a democracia, manipulando a opinião do leitor. Para analisar os padrões dessas notícias foram utilizadas as técnicas de pré-processamento Sumarização e *Word2Vec*. O objetivo desse trabalho é apresentar um modelo capaz de classificar textos entre *Fake News* e notícias reais usando *Deep Learning* e Aprendizado de Máquina Tradicional. No campo de *Deep Learning* foram utilizadas as arquiteturas de Redes Neurais Recorrentes e *Long Short-Term Memory*, enquanto que no Aprendizado de Máquina Tradicional os algoritmos *Random Forest* e *Support Vector Machine*. Dos resultados obtidos, os experimentos com *Long Short-Term Memory* retornaram a maior acurácia, com o valor de 79.3%.

Palavras-chave: Aprendizado Profundo. Notícias Falsas. Mineração de Texto.

MARUMO, FABIANO SHIITI. **Deep Learning to classification of Fake News by Text Summarization**. 2018. 54p. – State University of Londrina, Londrina, 2018.

ABSTRACT

News has always been important for society, because through them information is provided to people, whether by physical or digital means. In a society where citizens have the right to access information, the spread of false news has become a concern. Known by Fake News, the malicious use of information can compromise democracy by manipulating the reader's opinion. In order to analyze the standards of these news were used the preprocessing techniques Summarization and Word2Vec. The purpose of this work is to present a model capable of classifying texts between Fake News and real news using Deep Learning and Traditional Machine Learning. Deep Learning field were used the Recurrent Neural Networks and Long Short-Term Memory architectures, while in Traditional Machine Learning the Random Forest and Support Vector Machine algorithms. From the results obtained, the experiments with Long Short-Term Memory returned the highest accuracy, with a value of 79.3 %.

Keywords: Deep Learning. Fake News. Text Mining.

LISTA DE ILUSTRAÇÕES

Figura 1 – Diagrama do processo de mineração de texto, imagem baseado em [1] . 26
Figura 2 – Classificação de problema linearmente separável [2] 31
Figura 3 – Representação de uma Rede Neural Artificial 32
Figura 4 – Fluxograma em alto nível de diferentes áreas da IA, baseado e traduzido de [3]. 34
Figura 5 – Arquitetura da RNN [4] 35
Figura 6 – Arquitetura da LSTM [5] 36

LISTA DE TABELAS

Tabela 1 – Sentença original e pré-processada 27
Tabela 2 – Informações das amostras 39
Tabela 3 – Exemplo de FN 40
Tabela 4 – Exemplo de notícia real 40
Tabela 5 – Exemplo de notícia real sumarizada 41
Tabela 6 – Quantidade de amostras 45
Tabela 7 – Resultados DL 45
Tabela 8 – Resultados IA tradicional 46
Tabela 9 – Tempo de treinamento 46
Tabela 10 – Amostra com classificação incorreta 47

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizado de Máquina
DL	*Deep Learning*
FN	*Fake News*
LSTM	*Long Short-Term Memory*
MLP	*Multilayer Perceptron*
RF	*Random Forest*
RNN	*Recurrent Neural Network*
RNA	*Redes Neurais Artificiais*
SVM	*Support Vector Machine*

SUMÁRIO

1 INTRODUÇÃO 23

2 FUNDAMENTAÇÃO TEÓRICA 25
2.1 *Fake News* 25
2.2 Mineração de texto 26
2.2.1 Pré-processamento de texto 26
2.2.2 Transformação de Dados 27
2.2.3 Mineração de dados 28
2.2.4 Avaliação 28
2.3 Aprendizado de Máquina 28
2.3.1 *Random Forest* 29
2.3.2 **Support Vector Machine** 30
2.3.3 Redes Neurais Artificiais 31
2.3.4 *Deep Learning* 33
2.4 Algoritmos de *Deep Learning* 35
2.4.1 *Recurrent Neural Network* 35
2.4.2 *Long Short-Term Memory* 36

3 DESENVOLVIMENTO 39
3.1 Base de Dados 39
3.2 Frameworks utilizados 40
3.3 Pré-processamento 41
3.4 Mineração de Dados 42
3.5 Métricas de avaliação 42

4 RESULTADOS 45

5 CONCLUSÃO E TRABALHOS FUTUROS 49

REFERÊNCIAS 51

1 INTRODUÇÃO

Diante de uma sociedade onde as trocas de informação se tornaram meios práticos e de fácil acesso, o uso de notícias falsas vem gerando diferentes problemas, como influências na escolha de um candidato no período de eleições[6]. É importante salientar que o uso desse tipo de recurso pode comprometer uma sociedade democrática.

Segundo [7], os jovens atualmente tendem a consumir menos os meios de comunicação mais antigos, como jornais, rádio e TV, pois acreditam que, além de serem desinteressantes, são repetitivos. Assim eles acabam utilizando as redes sociais como principal meio de informação, acarretando em jovens mais alienados em assuntos que não são de seu interesse.

As notícias publicadas em redes sociais e aplicativos de troca de mensagens possuem um baixo nível de confiabilidade [8], causado principalmente pela falta de filtragem de *Fake News* (FN) ou a inexistência dela. Ainda assim as FN conseguem atingir um público que "quer acreditar nelas, as consome mesmo que suspeite delas, pois quer ver sua ideologia e seus preconceitos confirmados"[9]. Recentemente, o *Facebook* e a *Google* buscam enfrentar o problema de FN após receber várias críticas onde apontam que as as notícias falsas que circularam em seus sites podem ter influenciados as eleições presidenciais [10].

De acordo com [9], as FN ajudarão no fortalecimento do jornalismo de qualidade, como aconteceu em um caso de FN muito divulgada no Brasil, onde tempos depois, alguns meios de comunicação, como Veja e G1, desmentiram esses fatos [11]. Apesar das FN alcançar um maior número de pessoas que as notícias verdadeiras [12], houve um aumento no consumo de jornais nas plataformas digitais nos últimos anos [13].

Na França, nas vésperas da eleições presidenciais de 2017, foi criado o site "CrossCheck", com o intuito de combater as FN no país, realizando a checagem de fatos. No Brasil existem também os sites "Aos Fatos"[1] e "Truco - Agência Pública"[2], que possuem a mesma função que o site francês. Outros jornais, como "O Globo"[3], "Folha de São Paulo"[4] e "G1"[5], possuem o seu próprio site para essse mesmo propósito [14].

O site "Aos Fatos", por exemplo, realiza posteriormente a checagem de fatos selecionando uma notícia à partir de sua relevância, consultando sua fonte original para checar a veracidade, fazendo validações com fontes confiáveis e oficiais. Fontes alternativas também são utilizadas com a intenção de contrariar ou subsidiar os dados oficiais.

[1] https://aosfatos.org/
[2] https://apublica.org/checagem/
[3] https://www.globo.com/
[4] https://www.folha.uol.com.br/
[5] https://g1.globo.com/

Após as validações com as fontes pesquisadas, é realizada a contextualização da notícia e, em seguida, é classificada em uma das seis categorias: verdadeira, imprecisa, exagerada, falsa, contraditória ou insustentável [15].

Existem algumas possíveis propostas para detecção de FN em uma notícia online, como procurar pelas fontes de imagens e verificar em quais sites elas também foram utilizadas [16]. Outra alternativa é modificar o algoritmo de busca da *Google*, o *PageRank*, levando em consideração não apenas a quantidade de referências de sites citados, mas também suas relevâncias, como sua importância e confiança [17]. Além disso, é também viável utilizar a fonte, título, texto e vídeos das notícias como dados para classificação de FN [18]. A proposta para detecção de FN que foi escolhida para esse trabalho foi analisar os textos das notícias reais e de FN, buscando diferenciá-los através da estrutura do texto.

Para um ser humano ser capaz de lidar com vários dados de textos, seria necessário ser profissional na área, ter tempo para explorar cada notícia e classificá-la. Para resolver esse problema, neste trabalho será utilizada uma área da Inteligência Artificial chamada Aprendizado de Máquina (AM) [19] que possibilita a classificação de textos de forma rápida, podendo alcançar desempenho melhores que o próprio ser humano. Porém, devido a abstração do problema e a dificuldade de identificar quais parâmetros são os mais eficientes no seu aprendizado para classificação, o *Deep Learning* (DL), uma subárea de Aprendizado de Máquina, surge como um possível candidato para resolver esse desafio.

Como as FN possuem um padrão onde utilizam palavras-chaves mais populares, tanto em seus títulos como em seus textos, a sumarização [20] pode ser uma abordagem interessante de ser avaliada, visto que ela cria um pequeno sumário a partir de um texto fonte. A partir do sumário, algoritmos de DL poderiam associar FN ao uso de palavras-chave, diferente de uma notícia verídica, que foca em informar o leitor evitando repetir demasiadamente as mesmas palavras. Para auxiliar a associação das palavras na sumarização, o Word2Vec [21] foi o modelo escolhido para essa tarefa, visto que ele consegue associar um conjunto de palavras, verificando o contexto onde se encontram.

O objetivo deste trabalho é criar um classificador capaz de detectar FN em notícias na língua portuguesa. O seu foco é principalmente utilizar dois algoritmos de DL, *Long Short-Term Memory* (LSTM) e *Recurrent Neural Network* (RNN). A organização desse documento é feita da seguinte forma: a Seção 2 apresenta os conceitos teóricos do trabalho, a Seção 3 explica a metodologia, o experimento e os algoritmos utilizados durante o seu desenvolvimento. A Seção 4 descreve os resultados obtidos e, por último na Seção 5 é apresentado as conclusões do presente estudo.

2 FUNDAMENTAÇÃO TEÓRICA

Nessa seção são apresentados as definições teóricas que fundamentam o presente trabalho.

2.1 *Fake News*

O termo *Fake News* (FN) voltou a ser bastante utilizado após as eleições presidencias de 2016 nos Estados Unidos da América e pode ser entendido de maneira simples pelo sentido literal da palavra, notícias falsas. Porém ela pode ser definida de uma outra forma. De acordo com [22], para uma notícia ser classificada como FN ela deve possuir o conteúdo falso e sem embasamento, com o intuito de enganar e manipular o leitor.

Algumas características usadas em sites de FN por [23] foram que eles são registrados com domínio ".com"ou ".org", dificultando a identificação dos responsáveis com a mesma transparência que acontecem com os domínios registrados no Brasil, que terminam com ".br". Além disso, seus nomes são semelhantes a outros de jornais e de *blogs* autorais. Na parte do conteúdo do site as notícias não são assinadas e são cheias de opiniões, além de possuir um *layout* poluído e repleto de propagandas. Geralmente não possuem a página de "Quem Somos", e quando possuem não deixam claro quem são os autores do site.

Segundo [24], o ato de espalhar FN já acontece desde o século VI, quando um historiador bizantino difamou o imperador da época entre outras pessoas. Assim como aconteceu na eleição do Presidente Trump nos Estados Unidos, na França às vésperas da Revolução Francesa era comum a circulação de FN, apesar da censura a imprensa na época. Porém, se formos fazer uma comparação entre os dias de hoje e alguns anos atrás, o volume produzido de FN antigamente era bem menor, assim como o volume de notícias verdadeiras [25].

Um dos possíveis meios de espalhamento de FN são *bots* [26], que tem como principal objetivo a monetização através de propagandas, conseguida quando os usuários visitam um certo site. Para isso, são criados textos chamativos que possuem palavras-chave populares, mesmo que seus conteúdos não façam sentido [27].

De acordo com [27], esses programas (*bots*) buscam por palavras-chave de maior destaque, registram os nomes de domínios baseados nessas palavras e criam contas de hospedagem para esse domínios.

Após concluído esse passo, são criados os sites com notícias falsas contendo palavras-chave e cheia de anúncios e finalizam carregando os sites para as contas de hospedagem nos seus respectivos domínios, e são referenciados com *links*, para aumentar a chance dos

sites serem buscados no *Google*, através do algoritmo de *PageRank*.

Além de *bots*, a propagação de FN também é feita por usuários que compartilham as informações sem ao menos utilizar do pensamento crítico para reconhecê-los, e também por jornalistas que possuem menos recursos, que acabam não verificando a veracidade da origem da notícia devido a pressa para a publicação [28].

2.2 Mineração de texto

Com a grande quantidade de dados em formatos textuais que são gerados e armazenados diariamente, houve uma demanda por técnicas de análise e extração de conhecimento à partir desses textos. A mineração de texto surgiu com esse propósito, buscando modelar padrões e extrair informações úteis e implícitas para facilitar o entendimento de documentos textuais, buscar informações específicas e analisar de forma quantitativa e qualitativa grandes volumes de textos [29].

Segundo [30], o processo de extrair conhecimento de dados e utilizá-los como estratégias ou decisões é chamado Descoberta de Conhecimento em Bancos de Dados (do inglês *Knowledge Discovery in Databases*), que por sua vez, pode ser dividida nas seguintes etapas: aquisição de dados, pré-processamento de dados, transformação, mineração de dados e avaliação. Esses passos devem ser feitos na sequência, pois cada etapa depende do resultado da etapa anterior. Nos próximos tópicos serão abordados essas etapas conforme mostra a Figura 1.

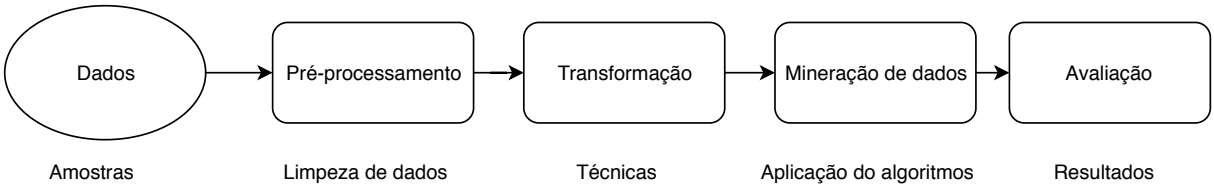

Figura 1 – Diagrama do processo de mineração de texto, imagem baseado em [1]

A primeira etapa, que se refere a aquisição de dados, consiste na seleção dos dados que representem o cenário real do problema estabelecido. Esse dados podem ser compostos de documentos, sites e e-mails por exemplo. Nesse trabalho os dados escolhidos foram retirados apenas de sites brasileiros.

2.2.1 Pré-processamento de texto

Segundo [1], os dados podem ser originalmente inadequados para serem processados pelos algoritmos de mineração. Por isso é necessário realizar o pré-processamento, que consistem em realizar uma etapa de limpeza no texto para garantir a qualidade dos dados, além de também diminuir o tempo de processamento.

Na Tabela 1, é mostrado um exemplo de pré-processamento de uma sentença. Na primeira linha se encontra o texto original, e na segunda o texto pré-processado. As colunas marcadas com "X"na linha "Pré-processada"indicam as palavras que foram removidas, como aconteceu nos dígitos de números ("2"), palavras com menos de três caracteres ("os"e "do"), caracteres não alfabéticos ("!") e espaçamentos múltiplos (“ ”). A única palavra que sofreu mudança foi "paulo", onde ocorreu a troca de letra maiúscula por minúscula.

Tabela 1 – Sentença original e pré-processada

Original	Os	2	gatos		do	Paulo	são	muito	bonitos	!
Pré-processada	X	X	gatos	X	X	paulo	são	muitos	bonitos	X

Ainda na parte de pré-processamento, é possível aplicar a Sumarização. Ela consiste na criação de um título ou resumo que capta as ideias principais de um artigo, não se limitando apenas a análise de algumas frases, mas verificando também o contexto principal [31].

Existem duas principais abordagens para construção dos resumos: *Extract* e *Abstract*. A abordagem utilizando *Extract* seleciona algumas passagens extraídas do texto analisado para a criação do seu resumo, enquanto a *Abstract* gera um novo texto analisando o texto de entrada [32].

É importante salientar que a parte de pré-processamento de texto também é de suma importância para modelagem do problema, pois ela pode acabar influenciando diretamente o resultado obtido [33].

2.2.2 Transformação de Dados

Para que os dados possam ser processados pelos algoritmos de AM, elas devem ser convertidas para um formato apropriado. Para isso, existem várias técnicas que realizam essa conversão.

A técnica escolhida para esse trabalho foi a *Word2Vec*, um método implementado de *Word Embedding* utilizando Redes Neurais Artificiais. O *Word Embedding*, por sua vez, é um tipo de representação de palavras que permite que palavras de significados parecidos tenham representações similares. Essas representações são vetores densos de altas dimensões, onde cada entrada é uma medida entre a associação entre a palavra e o contexto onde ela se encontra [21].

A *Word2Vec* possui 2 modelos para realizar o realizar o aprendizado das *Word Embedding*: *Continuous Bag-of-Words* e *Skip-Gram*.

O aprendizado do modelo *Continuous Bag-of-Words* é realizado predizendo uma palavra baseada no contexto de uma frase, enquanto o modelo *Skip-Gram* aprende pre-

vendo um contexto baseado em uma palavra escolhida. Em ambos modelos, o aprendizado é feito utilizando *back-propagation*, que é explicado posteriormente na subseção 2.3.3.

2.2.3 Mineração de dados

No processo de Mineração de Dados é escolhido o algoritmo que realizará a extração do conhecimento dos dados, levando em consideração o problema que deve ser resolvido. Esse processo pode ser repetido até que sejam alcançados resultados adequados [1].

Segundo [30], existem algumas tarefas nas quais as técnicas de Mineração de Dados podem ser aplicadas:

- Classificação: consiste na criação de um modelo que possa classificar dados que não estejam classificados.
- Regressão: utilizada para prever um valor de acordo com as variáveis .
- *Clustering* (Agrupamento): realiza a identificação de características dos dados, separando em grupos. É possível um dado possuir características de mais de um grupo diferente.

2.2.4 Avaliação

Após o fim do processo de mineração de dados, o resultado pode conter padrões irrelevantes ao usuário. Nesse sentido é utilizado diversas métricas de avaliação para auxiliar o entendimento desses padrões, como Precisão, Matriz de Confusão, *Recall* (Revocação) e Acurácia. Caso as avaliações não sejam satisfatórias, é possível repetir o processo todo até obter um resultado adequado.

2.3 Aprendizado de Máquina

Aprendizado de Máquina (AM) é definido por modificação ou adaptação em ações de um computador com o objetivo de tornar suas decisões ou previsões mais corretas, utilizando um conjunto de dados como forma de experiências para o seu processo de aprendizado [34]. O conjunto de dados usados para o processo de aprendizado são compostos por variáveis, que por sua vez são definidos como características ou atributos [35].

Tanto a qualidade quanto a quantidade das informações dos dados utilizados para o AM são relevantes para uma melhor performance e acerto de previsão [36].

Ao fim do processo AM, é gerado um algoritmo que pode ser classificado principalmente em dois tipos: supervisionado e não supervisionado. O supervisionado corresponde a tarefas em que o algoritmo tenta generalizar uma resposta utilizando como base um

conjuntos de dados com saída já rotuladas. Já no aprendizado não supervisionado, o objetivo é aprender padrões de características que são pertinentes em um conjunto de dados não classificados, agrupando por comportamentos [35].

Para [37], existem dois tipos de problemas que o aprendizado supervisionado tenta resolver: o primeiro é chamado de classificação. Ele ocorre em casos onde a saída esperada é um valor discreto, citando como exemplo a proposta desse trabalho, onde classificamos a saída entre duas categorias possíveis: FN ou notícia real. O outro problema a ser resolvido é conhecido como problema de regressão: onde os valores da saída são valores contínuos. Um exemplo seria prever o preço de uma casa, utilizando como entrada os valores da localização, tamanho do terreno e demanda de imóveis na região.

2.3.1 *Random Forest*

O *Random Forest* (RF) é uma algoritmo de AM que combina um conjunto de árvores de decisão, onde cada árvore é treinada utilizando um sub-conjunto de amostras selecionadas aleatoriamente e com repetição. A quantidade total de amostras utilizadas no sub-conjunto é o total de amostras do conjunto original de treinamento [38].

A Árvore de Decisão também é um algoritmo AM, que tem como finalidade criar um modelo em um formato de árvore. Cada nó dessa árvore contém uma questão, e as possíveis respotas para essa questão são os ramos que apontam para os nós filhos. Ao classificar um dado, é percorrido o caminho em direção da raíz até algum nó sem filho (folha), respondendo as perguntas de acordo com as características do dado. A folha determina a qual classe pertence um dado [39].

As duas principais medidas para avaliar as divisões nos nós são baseadas na entropia e Índice Gini.

A entropia visa calcular a impureza dos dados através da verificação da homogeneidade de um conjunto de dados em relação a sua classificação. Sua fórmula é dada pela Equação (2.1), onde S é um sub-conjunto do conjunto total de dados T e p_i é a probabilidade de um sub-conjunto de dados pertença a uma classe i dentre as possíveis M classes. O valor é menor quando um único p_i é um e os outros é zero, enquanto o valor maior acontece no momento em que todos os p_i são iguais.

$$H(S) = -\sum_{i=1}^{M} p_i.\log_2 p_i \tag{2.1}$$

Outra medida que calcula a impureza é o Índice de Gini, que mede o grau de heterogeneidade dos dados relacionado a cada nó. Sua representação é dada pela Equação (2.2), onde M é o total de classes. Quando o valor for zero significa que o sub-conjunto contém apenas dados de uma única classe, ou seja, é um nó puro.

$$GI(S) = 1 - \sum_{i=1}^{M} p_i^2 \tag{2.2}$$

Em árvores de classificação com partições binárias, o critério de Gini tende a isolar as folhas da classe mais frequente em um ramo, enquanto a Entropia balanceia o número de folhas em cada ramo [40].

O Ganho de Informação, representado pela Equação (2.3), é definida escolhendo uma medida de impureza I, como a Entropia ou Índice de Gini. O valor K na fórmula é o numero total de respostas do nó e S o conjunto de dados. Após calculado o Ganho de Informação, é escolhido o atributo que possui o maior valor para realizar a pergunta em um nó que dividirá árvore em ramos.

$$IG(S) = I(S) - \sum_{j=1}^{K} \frac{|Sj|}{|S|} I(Sj) \tag{2.3}$$

Quando a RF classifica um conjunto de dados, os dados percorrem todas as árvores criadas pela RF. Em problemas de regressão, o resultado é a média dos valores de saídas de cada árvore, enquanto em problemas de classificação, cada árvore vota e o elemento mais votado é a definição daquele conjunto de dados.

2.3.2 Support Vector Machine

A SVM é um algoritmo de AM supervisionado que cria um modelo que visa melhor separar as classes no espaço através de uma linha de separação, conhecido também como hiperplano. Ao mapear novos dados nesse espaço, eles são classificados de acordo da posição referente ao hiperplano. Na Figura 2, se mapearmos um dado a direita do hiperplano, o novo dado será da mesma classe de círculos pretos, e se for a esquerda, seria da classe dos círculos brancos.

Quanto maior a distância das classes desse hiperplano, melhor será o modelo obtido [41]. Na Figura 2 o hiperplano mostrado é considerado ótimo, pois ela consegue generalizar bens os dois tipos de amostras. Para auxiliar a criação desse hiperplano de separação foram utilizadas os vetores de suporte, criadas à partir da escolhidas das amostras mais próximas de cada classe.

Segundo [42], a SVM é adequada para classificação de texto, pois ela se lida bem com as características desse tipo de problema, como espaço de entrada de alta dimensão, poucas informações irrelevantes, vetor de documentos muito esparso e pela maioria dos problemas de classificação serem linearmente separáveis. Assim, de maneira geral, a SVM pode se sair bem para resolver a detecção de FN.

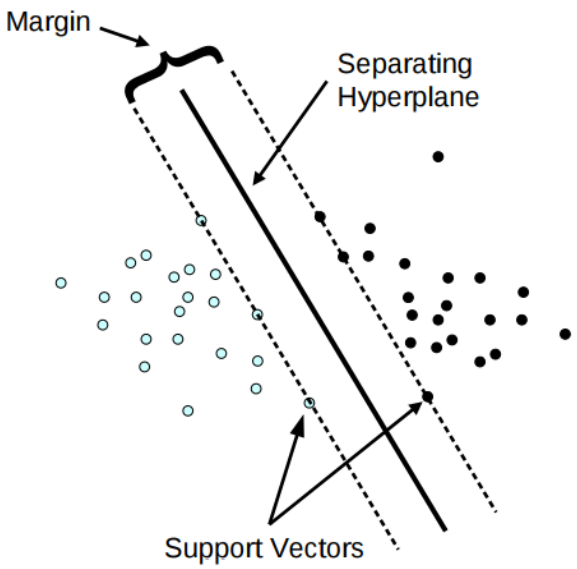

Figura 2 – Classificação de problema linearmente separável [2]

2.3.3 Redes Neurais Artificiais

Baseado na estrutura neural do cérebro, Redes Neurais Artificiais (RNA) são algoritmos computacionais. O primeiro modelo proposto foi chamado *Perceptron*, composto de apenas um neurônio. Esse modelo era apenas capaz de resolver problemas linearmente separáveis de classificação binária [43].

Matematicamente, os neurônios são representados pelas Equações (2.4) e (2.5). Na primeira equação, os valores de entrada X são multiplicadas pelos seus respectivos pesos sinápticos W, onde eles são somados com o viés (*bias*), que é utilizado para auxiliar nos ajustes dos pesos W. Sua saída V é então usada na Equação (2.5) como entrada para a função de ativação, que transformará os valores de saída.

$$v = \sum_{i=1}^{n}(w_i).(x_i) + bias \tag{2.4}$$

$$y = f(v) \tag{2.5}$$

Existem várias funções de ativações. Uma delas é a função de ativação sigmóide, que tem como valor de saída o intervalo de [0 , 1]. Usada na última camada da rede para classificar os valores para uma classe específica. Sua representação é dada pela Equação (2.6).

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.6}$$

Outra função de ativação é a Tangente Hiperbólica, representada como *tanh*, possui a saída entre os valores [-1 , 1]. Utilizada principalmente para classificação entre duas classes, além de poder mapear dados negativamente. Sua Equação (2.7) é dada pela divisão do seno hiperbólico pelo cosseno hiperbólico.

$$tanh(x) = \frac{senh(x)}{cosh(x)} = \frac{\frac{e^x - e^{-x}}{2}}{\frac{e^x + e^{-x}}{2}} \tag{2.7}$$

Utilizando vários neurônios em múltiplas camadas é possível resolver problemas não lineares [44]. Conhecido como *Multilayer Perceptron*, essa estrutura é composta por três camadas interconectadas: camada de entrada, camada de saída e as camadas ocultas (que ficam entre as camadas de entrada e saída). Essas camadas são representadas pela Figura 3, onde cada camada é composta por um ou mais neurônios, representados pelos círculos, e cada neurônio é totalmente conectada aos neurônios da próxima camada [43].

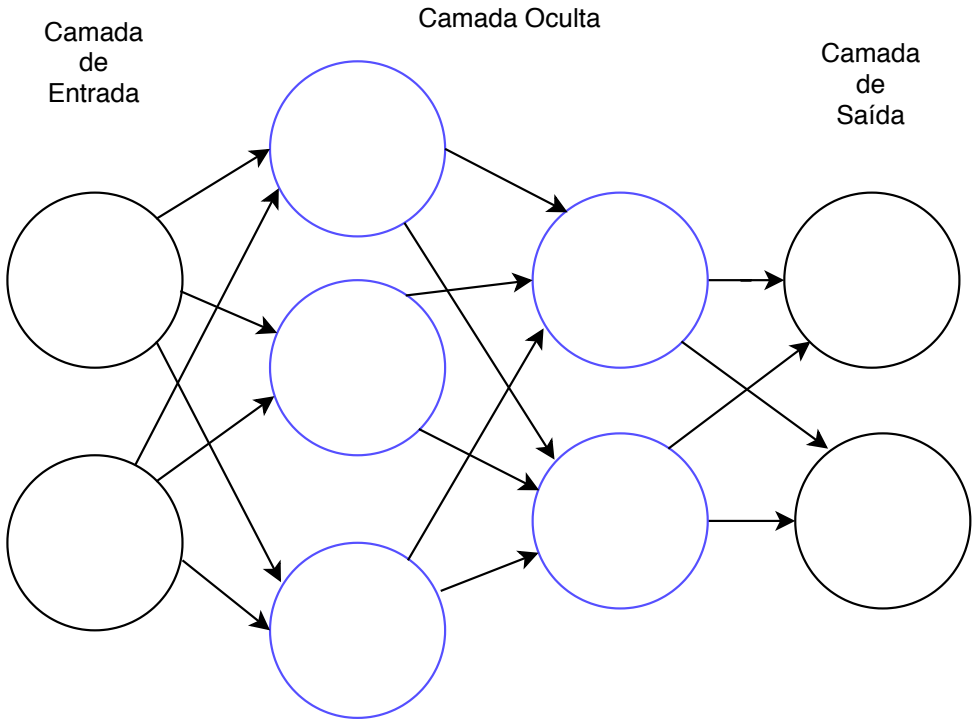

Figura 3 – Representação de uma Rede Neural Artificial

Um algoritmo supervisionado de treinamento é o *back-propagation*. Em sua primeira fase, para frente (*forward*), são gerados valores de saída da rede, e na segunda fase, para trás (*backward*), são utilizados esses valores de saída e os valores desejados para calcular os erros e atualizar os pesos sinápticos. Esses ajustes podem ser feitos utilizando algoritmos de otimização, como por exemplo o gradiente descendente [45] .

O método de gradiente descendente busca encontrar o valor de mínimo local, visto que este valor é diretamente proporcional ao valor do erro. Assim "o algoritmo *back-propagation* procura minimizar o erro obtido pela rede ajustando pesos e limiares para

que eles correspondam às coordenadas dos pontos mais baixos da superfície de erro. Para isto, ele utiliza um método de gradiente descendente."[45]

Os erros são calculados por funções de perda, que tem como objetivo escolher a forma como penalizar quando os resultados preditos diferem dos valores reais. Quanto maior o valor de erro, menor é o aprendizado do modelo.

2.3.4 *Deep Learning*

De acordo com [46], a *Deep Learning*(DL) é um sub-campo do AM que utiliza um conjunto de algoritmos buscando aprender diferentes níveis de abstração, utilizando várias camadas de processamentos não lineares.

Utilizando essas várias camadas de processamento não lineares, é possível realizar a detecção de padrões dos dados de entrada durante o seu processo de aprendizado. Em técnicas de AM tradicionais, esse processo de extração de padrões e de características é feito manualmente por um profissional da área. Na Figura 4 é representada essa ideia, onde algoritmos de AM tradicionais necessitam de mais trabalhos manuais para realizar a extração de descritores dos dados de entrada, se comparada ao DL, onde ela mesma faz essa extração. Os quadrados coloridos representam os passos onde o algoritmo realiza o seu processo de aprendizado [47].

O crescimento do uso de DL nos últimos anos foi proporcionado pelo aumento do poder computacional, com o avanço da unidade de processamento gráfico (GPUs), popularmente conhecida por placa de vídeo [48], visto que o tempo de processamento de uma DL é maior que a de um algoritmo de AM comum devido as suas várias camadas. Essa alta demanda computacional também deve-se ao DL utilizar uma grande quantidade de dados para o seu processo de aprendizado.

Visto que o DL realiza extração de descritores dos dados, ele poderia extrair características abstratas que as pessoas não conseguem descrever, e assim associar padrões nos textos de notícias reais, escritas por pessoas, e de FN, textos criadas por *bots* para fazer uma distinção entre os textos.

Atualmente, existem inúmeros *frameworks* na área de DL disponíveis, porém existem alguns que se destacaram e acabaram se tornando mais populares [49].

1. *Tensorflow*[1] : Considerado um dos melhores, se não o melhor, entre todos os *frameworks*, ele é muito popular em grandes empresas, como *Twitter* e *Google*, o que ocasiona em um maior número de materiais disponíveis na Internet. Possui uma documentação detalhada e um sistema de arquitetura flexível, com capacidade de ser utilizado nas plataformas *desktop* e *mobile*.

[1] https://www.tensorflow.org/?hl=pt-br

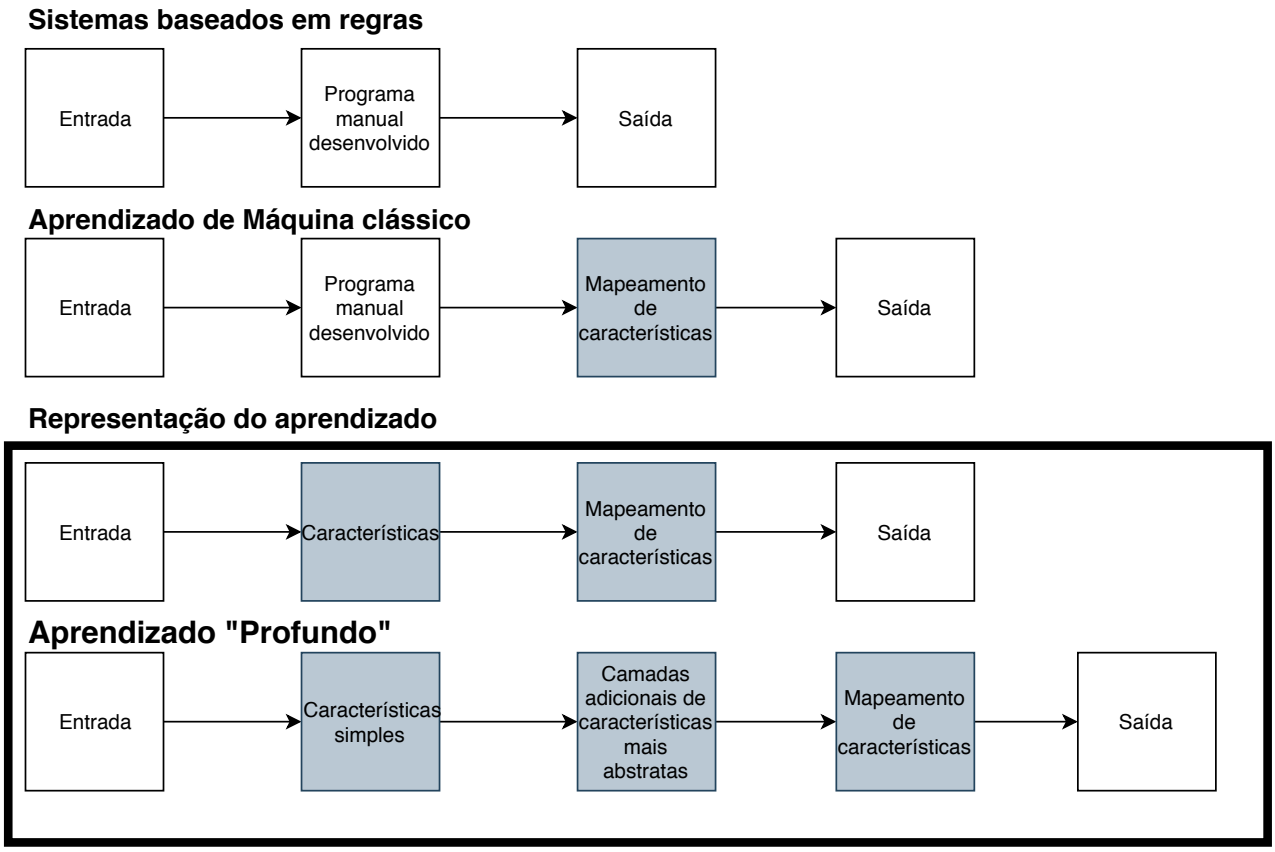

Figura 4 – Fluxograma em alto nível de diferentes áreas da IA, baseado e traduzido de [3].

2. *Caffe*[2]: Conhecido por ser um *framework* rápido, ele é recomendável para ser utilizado em tarefas de reconhecimento de visão, porém, sofre pela arquitetura utilizada, dificultando o desenvolvimento de redes mais complexas.

3. *The Microsoft Cognitive Toolkit* (CNTK)[3]: Popular por conseguir operar em várias máquinas e ter resultados melhores que a *Tensorflow*, ele é caracterizado por ser fácil de treinar e pela capacidade de combinar modelos populares entre os servidores.

4. *Pytorch*[4]: Considerado rival do *Tensorflow*, ele também é utilizado por grandes empresas, como *Facebook* e *Google*. O *Pytorch* consegue executar modelos complexos e possui um grande suporte para algoritmos de AM.

5. *MXNet*[5]: Ele é o único que possui um grande leque de opções de linguagem de programação para implementação, além de ser desenvolvido com o propósito de alcançar alta eficiência, produtividade e flexibilidade.

[2] http://caffe.berkeleyvision.org/
[3] https://www.microsoft.com/en-us/cognitive-toolkit/
[4] https://pytorch.org/
[5] https://mxnet.apache.org/

6. *Chainer*[6]: Focado mais para uso em reconhecimento de fala e análise de sentimento, o *Chainer* é poderoso, intuitivo e dinâmico, possibilitando a modificação das redes durante o tempo de processamento, um diferencial entre os seus concorrentes.

7. *Keras*[7]: Utilizado juntamente com o *Tensorflow*, a biblioteca dele é leve e principalmente usada na área de texto.

8. *Deeplearning4j*[8]: Dentre todos os citados, é o único desenvolvida em Java. É amplamente adotada na área comercial e industrial.

Para esse trabalho foram escolhidos os *frameworks Tensorflow*, devido a sua facilidade de uso, proporcionadas por um grande número tutorias online e de documentações, e o *Keras*, que possui vários algoritmos desenvolvido que auxiliam na construção no modelo de DL.

2.4 Algoritmos de *Deep Learning*

Foram escolhidos dois algoritmos de aprendizado que serão utilizados no DL para realizarem a extração de conhecimento dos dados.

2.4.1 *Recurrent Neural Network*

A RNN é uma RNA recursiva, onde os valores da entrada são dependentes do passos que aconteceram nos neurônios anteriores. Sua ideia está representada na Figura 5, onde as RNA A recebem as entradas x, produzem a saída h, que também são utilizada para alimentar as entradas das próximas camadas ocultas. Com a alimentação da saída anterior (h_{t-1}) na etapa atual (h_t) é simulada uma memória dos resultados passados, criando uma dependência das sequências dos dados.

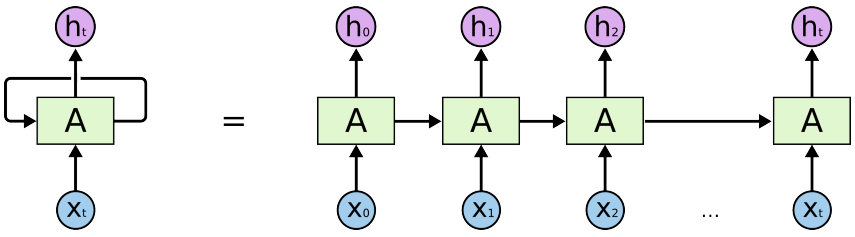

Figura 5 – Arquitetura da RNN [4]

De acordo com [50], um problema que ocorre na RNN é o desaparecimento de gradiente. Como a atualização dos pesos é resultado da multiplicação do termo de erro da camada anterior e a entrada da camada atual, onde termo de erro para uma determinada

[6] https://chainer.org/
[7] https://keras.io/
[8] https://deeplearning4j.org/

camada é o produto de todos os erros das camadas anteriores. Ao lidar com funções de ativação como a função sigmóide, os pequenos valores de suas derivadas, que ocorrem na função do cálculo de erro, são multiplicados várias vezes à medida que nos aproximamos em direção às camadas atuais. Como resultado, o gradiente quase desaparece, dificultando o treinamento dessas camadas.

Esse problema de desaparecimento de gradiente acaba influenciando a RNN, limitando a capacidade de armazenamento ao lidar com sequências de dados, conseguindo lidar com dados de curto prazo mas não com de longo prazo [50].

2.4.2 *Long Short-Term Memory*

Uma rede LSTM é composta de RNN com unidades LSTM. E utilizando essas unidades, segundo [4] e [5], as redes LSTM conseguem ter uma memória de longo prazo mais eficiente que RNN ao criar uma célula de estado interno de memória que soma a entrada processada ao invés de multiplicá-la, reduzindo o problema de pequenos gradientes. Além disso, ela possui um conceito chamado *forget gate*, que determina quais estados devem ser relembrados ou esquecidos utilizando as entradas anteriores. A Figura 6 está representado o diagrama da LSTM.

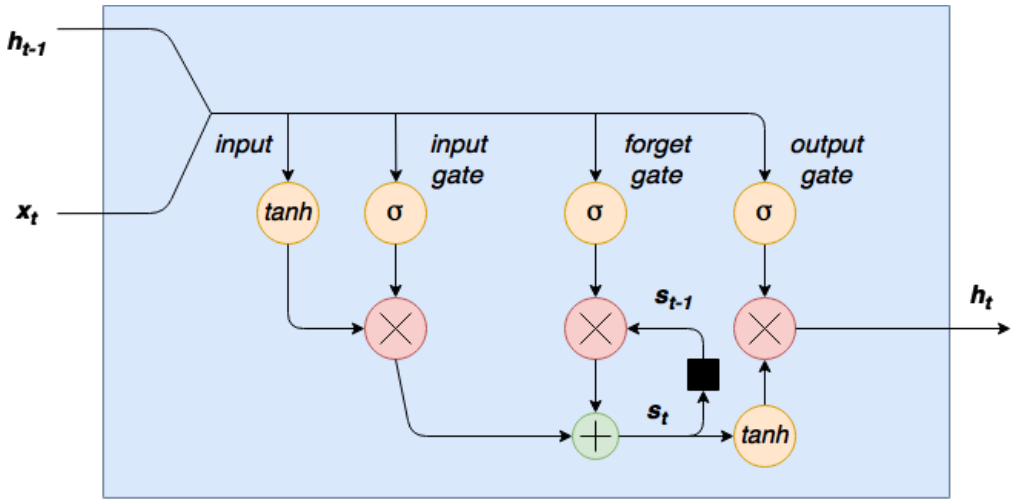

Figura 6 – Arquitetura da LSTM [5]

O conjunto de entrada, que será representado pela letra z, é determinado pela concatenação da entrada atual x e a saída da célula anterior h. A LSTM é dividida em três estágios:

- Estágio *input gate*: As entradas z são utilizadas pela função de ativação tangente hiperbólica, que limita a saída aos valores entre [-1 , 1]. Na fase de *input gate*, a função de ativação sigmóide utiliza as entradas z, que geram valores de saídas 0 ou 1. A saída final desse estágio é a multiplicação desses dois resultados anteriores.

- Estágio *forget gate*: Nesse estágio temos a variável s, que representa o estado de célula interna de memória. A saída da função sigmóide na entrada z será multiplicada pela célula de memória *s*, que armazena os dados aprendidos do passos antecessores, determinando quais serão os novos dados a serem lembrados. Isso permite a célula LSTM aprender de acordo com o contexto. A sua saída é somada com o resultado do estágio de *input gate*, ao invés de multiplicada, assim reduzindo o problema de desaparecimento de gradiente enfrentado pela RNN, e será armazenada na célula de memória s, para que ela possa ser utilizada para a próxima rede LSTM.

- Estágio *output gate*: A saída final desse modelo LSTM será o resultado da multiplicação da entrada z passada pela função de ativação sigmóide com a saída do estágio *forget gate* passada pela função de ativação tangente.

Vale ressaltar que as operações de multiplicação feitas pelas matrizes que aparecem na Figura 6, representadas por X, é o produto de Hadamard [51], onde cada elemento de uma matriz é multiplicado pelo elemento de mesma posição da matriz multiplicante.

Com a capacidade de conseguir guardar dados de longo prazo na memória, a LSTM pode associar padrões de textos onde há muitas repetições de palavras-chaves com FN, facilitando a sua identificação.

3 DESENVOLVIMENTO

Nesta seção são detalhados os métodos e algoritmos que foram utilizados para o desenvolvimento desse trabalho.

3.1 Base de Dados

Para se aproximar o máximo possível do cenário real, foram utilizadas notícias que abordam temas variados, como saúde, justiça, política, economia, todas de domínios públicos brasileiros, no idioma português. As amostras podem ser divididas em dois tipos:

1. Notícias verdadeiras: Os dados foram retirados do site de notícias da Universidade Estadual de Londrina, Agência UEL de notícias[1], coletadas durante o período de 2017 a 2018, juntamente com o site G1[2].

2. *Fake News*: Para coletar diferentes modelos de notícias falsas, foram usados os domínios públicos "Diário de Pernambuco[3]"e o "Sensacionalista[4]", estas coletadas com uma maior diferença de tempo, possuindo algumas que datam mais de 10 anos desde sua publicação.

Algumas estatísticas foram retiradas das amostras, como mostra a Tabela 2. Os dados mostram que as notícias reais apresentam um maior número de palavras e sentenças se comparadas as FN. Os textos das Tabela 3 e Tabela 4 mostram, respectivamente, um exemplo de FN e de notícia real que pertencem a base de amostras.

Tabela 2 – Informações das amostras

	Fake News	Real	Média
Média de palavras por notícia	184	288	236
Média de sentenças por notícia	9.74	14.64	12.19

Afim de realizar o balanceamento na quantidade de notícias, foi utilizada a técnica de sub-amostragem(*undersampling*) aleatório, que consiste na retirada aleatória de amostras da classe de maior quantidade. Assim o número de notícias para cada classe foi modificado para 286.

[1] https://g1.globo.com
[2] http://www.uel.br/com/agenciaueldenoticias
[3] http://www.diariodepernambuco.com.br
[4] https://www.sensacionalista.com.br

Tabela 3 – Exemplo de FN

Tipo de Notícia	FN
Conteúdo	A prefeitura do Rio de Janeiro anunciou hoje que as passagens na cidade passarão a custar R4.,$Comisso, oscariocaspassarãoa$ ser oficialmente assaltados. O prefeito Marcelo Crivella explicou que como muitos cidadãos saem de casa com medo de ser assaltados, a nova medida vai trazer paz. “Já será certo que assim que você entrar no ônibus você já vai perder a carteira”, disse um assessor.,O novo preço despertou esperança nos passageiros. “Por R4$talvezagentetenhacafezinhoechurroscomNutella$, $disse$ um deles.

Tabela 4 – Exemplo de notícia real

Tipo de Notícia	Notícia real
Conteúdo	O Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), em parceria com o Instituto Euvaldo Lodi (IEL), e com o apoio do Fórum Nacional de Gestores de Inovação e Transferência de Tecnologia (FORTEC), divulgam a Chamada Pública para o Programa INOVATEC. A Chamada é fruto de acordo da parceria firmado entre o IEL e o CNPq, cujo objetivo é a implantação do Programa INOVATEC. Tal Programa pretende fomentar a participação de estudantes de graduação em projetos de pesquisa, desenvolvimento e inovação PD&I de interesse do setor empresarial, em parceria com instituições de ensino superior e empresas. O Programa concederá até 200 bolsas de Iniciação Tecnológica e Industrial (ITI) do CNPq para estudantes regularmente matriculados em curso superior ou superior tecnológico, além de auxílios à pesquisa aos professores (coordenadores de projeto). As empresas serão responsáveis pelo custeio integral dos auxílios aos projetos, no valor de R$ 4.800, para despesas de custeio e capital. As propostas podem ser submetidas até 15 de dezembro deste ano, por meio do endereço http://www.portaldaindustria.com.br/iel/canais/inovatec/. (Com informações da Coordenação de Comunicação Social do CNPq)

3.2 Frameworks utilizados

A linguagem de programação utilizada foi o *Python* versão 3.6.6 64 bits [52]. Para o pré-processamento, foram utilizadas os *frameworks Gensim*[53] versão 3.5 e *Natural Language Toolkit* versão 3.4, que possuem algoritmos criados com foco em textos, e para modelagem do problema foram utilizadas os *frameworks* de aprendizado de máquina *Tensorflow* versão 1.12 [54] juntamente com o *Keras*[55] versão 2.2.4 e o *scikit-learn* versão 0.20.2.

3.3 Pré-processamento

O processo de pré-processamento foi feito respectivamente na ordem em que cada etapa é citada. Primeiro foi trocado as letras maiúsculas por minúsculas utilizando *Python*, em seguida foram removidos dígitos de números, utilizando *Gensim*, e palavras vazias (*stop words*), utilizando *Natural Language Toolkit*. Foi realizado a stemização(*stemming*) utilizando o código[1]. Após isso, foram feitos a sumarização, remoção de palavras não-alfabéticos, troca espaços múltiplos por espaço único e remoção de palavras com 2 caracteres ou menos utilizado novamente *Gensim*.

Na etapa de sumarização, o algoritmo do *Gensim* (que utiliza a abordagem *Extract*) não consegue processar textos de apenas uma sentença, gerando erro. Nesses casos foi decidido manter os textos originais como textos sumarizados, visando deixar a maior quantidade de amostras. Um exemplo de resultado onde a sumarização é aplicada no texto da Tabela 4 é mostrado na Tabela 5, onde o algoritmo escolheu a primeira sentença como o resumo da noticia.

Tabela 5 – Exemplo de notícia real sumarizada

Tipo de Notícia	Notícia real
Conteúdo	conselho nacional desenvolvimento científico tecnológico cnpq parceria instituto euvaldo lodi iel apoio fórum nacional gestores inovação transferência tecnologia fortec divulgam chamada pública programa inovatec

O resultado após esse processo de limpeza de texto é a geração de dois tipos de amostras: textos sumarizados e textos pré-processados, sendo este último obtido seguindo exatamente o mesmo processo descrito acima de pré-processamento, apenas omitindo a etapa de sumarização. Cada uma desses tipos de amostras são utilizadas para a aplicação do algoritmo de *Word2vec* do *framework Gensim*, gerando o seu próprio vocabulário. Os parâmetros escolhidos para o *Word2vec* foram a utilização do modelo *Skip Gram*, que segundo o autor do *Word2Vec* funciona melhor para uma pequena quantidade de dados, dimensionalidade dos vetores que representam as palvras com tamanho 10 e a escolha de não utilizar palavras com frequência menor que 2. Os parâmetros restantes foram utilizados com seus valores padrões.

Cada palavra no vocabulário do *Word2Vec* foi transformado em um vetor de 10 posições para 1 posição através da soma dessas 10 posições, vista que a adição é uma boa estimativa de composição semântica. Esses valores são usados para substituir as palavras das amostras pelos seus respectivos valores, servindo como entrada para os algoritmos de AM. Em casos onde as palavras não existissem no vocabulário do *Word2Vec*, foi atribuído o valor 0 para elas.

[1] https://www.nltk.org/_modules/nltk/stem/rslp.html

3.4 Mineração de Dados

A fase de uso de AM pode ser descrita de acordo com os seguintes passos:

1. Realizar o balanceamento das amostras, deixando uma quantidade igual para cada tipo de amostra, ou seja, para cada notícia real existe uma FN.
2. Realizar a validação cruzada, usando *holdout*: São separadas as amostras em dois subconjuntos, um para treinamento, que serão utilizadas para moldar o problema, e outro para teste, que serão classificadas pelo modelo criado anteriormente. Foi decidido usar 80% para o primeiro caso e 20% para o segundo.
3. Definir a arquitetura de DL, escolhendo as camadas da rede, utilizando o *Keras*.
4. Definir os valores dos hiperparâmetros que serão utilizados nos algoritmos.
5. Aplicar os algoritmo de classificação de IA tradicional, como RF e SVM, e DL, como RNN e LSTM , utilizando as amostras de treinamento para a criação do modelos de predição, utilizando 20 épocas.
6. Classificar as amostras de teste, utilizando o modelo gerado.

Na etapa 5, o algoritmo de RF foi definido com 100 árvores na floresta e o algoritmo SVM utilizou o kernel linear. O restante dos outros parâmetros foram utilizados em seus valores padrões. Ainda na etapa 5, a arquitetura da LSTM é composta de 3 camadas: uma camada de entrada, uma camada bidirecional de LSTM com saída de dimensionalidade 100, e uma camada densa com saída de dimensionalidade 2 com função de ativação sigmóide. A função de perda escolhida foi *binary crossentropy*, visto que ela é utilizada para problemas de classificação binária, e o otimizador escolhido foi *RMSProp*, uma recomendação de otimizador pelo próprio *Keras* para RNNs. A estrutura bidirecional consite na junção de duas RNN independentes, o que permite com que as redes tenham informações de trás e da frente sobre a seqüência a cada etapa de tempo. Para a arquitetura da RNN foi escolhida a mesma rede da LSTM, substituindo apenas a camada LSTM por RNN.

3.5 Métricas de avaliação

A métrica de avaliação será composta de quatro classificações:

- Verdadeiro positivo : Análise e texto são FN.
- Verdadeiro negativo : Análise e texto não são FN.
- Falso positivo : Análise não é FN e texto é FN.

- Falso negativo : Análise é FN e texto não é FN.

Para avaliar os resultados obtidos foram utilizados as métricas acurácia, precisão e revocação (*recall*). Suas fórmulas se encontram nas Equações (3.1) (3.2) (3.3), respectivamente. A acurácia visa medir a frequência de acertos, a precisão mede o grau de acerto quando os conjuntos são preditos como a classe a ser resolvida. Neste trabalho, a precisão é interpretado como quantidade de acertos dos dados classificados como FN, e a revocação calcula a frequência de acerto quando a classe do problema é classificada como classe do problema, que neste trabalho, pode ser interpretado como frequência de classificação de FN quando o texto é FN. Para ter uma maior confiabilidade, cada resultado foi obtido a partir da média após o algoritmo ser executado dez vezes.

$$Acurácia = \frac{VerdadeiroPositivo + VerdadeiroNegativo}{Total} \tag{3.1}$$

$$Precisão = \frac{VerdadeiroPositivo}{VerdadeiroPositivo + FalsoPositivo} \tag{3.2}$$

$$Revocação = \frac{VerdadeiroPositivo}{VerdadeiroPositivo + FalsoNegativo} \tag{3.3}$$

4 RESULTADOS

Houveram casos onde a aplicação da sumarização não gerou nenhum sumário, em média ocorrida na maioria da vezes em textos de FN, sendo 65% de sumários criados com sucesso para FN e 82% para notícias reais, como mostra a Tabela 6

Tabela 6 – Quantidade de amostras

	Fake News	Real	Total
Textos Pré-processados	286	513	799
Textos Sumarizados	187	422	609
Sumarização com sucesso	0.65	0.82	0.76

Como é mostrado nos resultados da Tabela 7 e Tabela 8, o melhor desempenho foi obtido pela LSTM em textos pré-processados com acurácia de 79.3% seguida da RF em textos pré-processados com 78.2% e SVM em textos pré-processados com 67.2%.

Tabela 7 – Resultados DL

	LSTM		RNN	
	Pré-processado	Sumarizado	Pré-processado	Sumarizado
Verdadeiro Positivo(%)	**77**	59	24	39
Verdadeiro Negativo(%)	**81**	52	79	55
Falso Positivo(%)	**19**	48	21	61
Falso Negativo(%)	**23**	41	74	45
Acurácia (%)	**79.3**	55.9	52.5	47.3
Precisão	**0.805**	0.555	0.555	0.468
Revocação	**0.772**	0.594	0.258	0.394

A LSTM, pela sua característica de possuir controle de longa memória, mostrou lidar bem quando o texto possui uma maior quantidade de dados, como acontece nos textos pré-processados, obtendo um valor equilibrado na predição de FN e notícia real, com a acurácia de 79.3%. Nas notícias sumarizadas, como os textos são transformados em resumos, a LSTM não teve dados suficientes para poder diferenciar as estruturas de texto de cada notícia, o que acarretou em uma acurácia menor com o valor de 55.9%.

Nos resultados da RNN, a maior de taxa de acerto ocorreram na predição de notícias reais, com 79% em textos pré-processados e 55 em textos sumarizados. Vale destacar também que em textos sumarizados o RNN teve um aumento de 15% de acerto para FN se comparada a texto pré-processados. Porém teve um diminuição de 24% no acerto de notícias reais.

Nos resultados dos algoritmos de IA tradicional, as duas maiores acurácias ocorreram em textos pré-processados, mostrando que tanto a RF como a SVM tiveram melhores

Tabela 8 – Resultados IA tradicional

	RF		SVM	
	Pré-processado	Sumarizado	Pré-processado	Sumarizado
Verdadeiro Positivo(%)	**74**	56	69	67
Verdadeiro Negativo(%)	**82**	69	66	48
Falso Positivo(%)	**18**	31	34	52
Falso Negativo(%)	**26**	44	31	33
Acurácia (%)	**78.2**	62.8	67.2	57.5
Precisão	**0.807**	0.648	0.667	0.562
Revocação	**0.743**	0.563	0.689	0.671

desempenhos analisando uma maior quantidade de dados. Analisando os resultados da tabelas de DL e de IA, a RF foi a única que teve resultados que quase se igualaram a LSTM em textos pré-processados. Porém, em textos sumarizados, vemos que a RF obteve a maior acurácia, com 62.8%, seguida de outro algoritmo de IA, a SVM. A LSTM alcança apenas o terceiro lugar, com 55.9%.

Tabela 9 – Tempo de treinamento

	Tempo por época	Éoocas	Tempo Total
RNN - Pré-processado	4s	20	80s
RNN - Sumarizado	1s	20	20s
LSTM - Pré-processado	25s	20	500s
LSTM - Sumarizado	7s	20	140s
RF - Pré-processado	0.22s	**1**	0.22s
RF - Sumarizado	0.16s	**1**	0.16s
SVM - Pré-processado	0.18s	**1**	0.18s
SVM - Sumarizado	**0.03s**	**1**	**0.03s**

Ao verificarmos os dados da Tabela 9, vemos uma grande diferença no tempo de treinamento entre os algoritmos de IA e das redes de DL. Enquanto os algoritmos de IA demoram menos de um segundo para realizar o treinamento do algoritmo inteiro, nas redes de DL o treinamento em apenas uma época demorou pelo menos 1 segundo, na ocasião mais rápida. O tempo total variou de 20 segundos a 500 segundos para treinar totalmente as redes de DL.

Nos classificadores de maior acurácia (DL e RF em textos pré-processados) houveram predições incorretas nas mesmas amostras. Uma delas foi o texto que se encontra na Tabela 10, retirada do site "Sensacionalista". Verificando suas informações temos 247 palavras e 12 sentenças, o que difere dos valores apresentados na Tabela 2, onde média de palavras e sentenças de um texto de FN são inferiores. Porém elas se aproximam dos valores da média das amostras, o que pode ter influenciado na sua classificação incorreta.

Tabela 10 – Amostra com classificação incorreta

Tipo de Notícia	FN
Conteúdo	O ministro da Saúde José Gomes Temporão disse hoje que preservativos devem ser usados “até para comer bananas” na África do Sul. Temporão respondia a uma pergunta sobre que cuidados o turista brasileiro deveria tomar caso viajasse para aquele país.,Temporão lembrou que a aids é um grande problema de saúde pública na África do Sul. Há pouco mais de um mês o ministro afirmou que “fazer sexo cinco vezes ao dia” é o melhor remédio contra a hipertensão.,A fala do ministro pegou muito mal em todos os setores da sociedade. Até a Associação Brasileira de Fomento à Banana (ABFB) endereçou uma nota de repúdio ao Ministério da Saúde afirmando que “esse tipo de comentário pretensamente jocoso cria confusão na cabeça das pessoas”. A ABFB anunciou que já constata queda na venda da penca de banana. “Sem falar que é praticamente impossível comer uma banana enquanto esta está envolta num preservativo”, concluiu o texto.,A maior reclamação, no entanto, veio do portavoz do presidente sulafricano Jacob Zuma. “O que ele tem contra as bananas do nosso país?”, disse Paul Madado, portavoz da presidência. “Eu mesmo já comi coisa muito pior no Brasil”., A crise diplomática exigiu que Lula interrompesse sua agenda e saísse em defesa de Temporão: “O ministro fala a língua do povo. Eu mesmo falei para os garotos da Seleção nem levarem suas gravatas do sexo para a África, porque a coisa lá está feia”.,Dunga, o técnico da seleção, mimimizou o problema: “Nem todo mundo gosta de sexo

5 CONCLUSÃO E TRABALHOS FUTUROS

A preocupação com as consequências que as FN podem causar na sociedade, como influências em campanhas eleitorais, foi a inspiração deste trabalho para realizar a detecção de FN. O método proposto foi testar o DL, utilizando RNN e LSTM e e algoritmos de IA, como SVM e RF, para a criação de modelos capazes de aprenderem padrões de textos pré-processados e sumarizados de FN.

O LSTM obteve o melhor resultado geral, com acurácia média de 79.3% em textos pré-processados. Em textos sumarizados, a RF alcançou a maior acurácia, com 62.8%. Se considerarmos o tempo de treinamento dos algoritmos, a RF seria considera melhor que a LSTM, visto que apesar de que os resultados da LSTM foram levemente superiores, o tempo gasto pela RF foi muito inferior.

As redes de DL mostraram um grande pontencial, visto que neste trabalho foi utilizada uma rede não muito complexa. Seria interessante utilizar uma arquitetura de DL mais complexa para aprimorar os resultados, além de aumentar a quantidade de amostras com as mais diversas fontes de notícias, permitindo aos algoritmos novas descobertas de padrões de texto das FN.

REFERÊNCIAS

[1] MORAIS, E. A. M.; AMBRÓSIO, A. P. L. Mineração de textos. *Relatório Técnico–Instituto de Informática (UFG)*, 2007.

[2] MEYER, D.; WIEN, F. T. Support vector machines. *R News*, v. 1, n. 3, p. 23–26, 2001.

[3] GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. 1–10 p. <http://www.deeplearningbook.org>.

[4] OLAH, C. *Understanding LSTM Networks*. 2015. Disponível em <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>.

[5] ANDY. *Recurrent neural networks and LSTM tutorial in Python and TensorFlow*. 2017. Disponível em <http://adventuresinmachinelearning.com/recurrent-neural-networks-lstm-tutorial-tensorflow/>.

[6] 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.

[7] MARCHI, R. With facebook, blogs, and fake news, teens reject journalistic "objectivity". *Journal of Communication Inquiry*, SAGE Publications Sage CA: Los Angeles, CA, v. 36, n. 3, p. 246–262, 2012.

[8] NEWMAN, N. et al. Reuters institute digital news report 2017. 2017.

[9] HERMIDA, X. *"Fake news' tornam o jornalismo de qualidade mais necessário do que nunca", diz diretor do EL PAÍS*. 2018. Disponível em <https://brasil.elpais.com/brasil/2018/02/20/actualidad/1519128638_053924.html>.

[10] WINGFIELD, N.; ISAAC, M.; BENNER, K. Google and facebook take aim at fake news sites. *The New York Times*, v. 11, p. 12, 2016.

[11] NETTO, C. F. W. L.; PERUYERA, M. S. Fake news como ferramenta de propaganda política na internet1.

[12] AFP. *Meios de comunicação contra-atacam as fake news*. 2018. Disponível em <https://exame.abril.com.br/mundo/meios-de-comunicacao-contra-atacam-as-fake-news/>.

[13] ZIRONDI, M. *Jornais investem em assinaturas digitais e captam novos leitores*. 2017. Disponível em <http://propmark.com.br/midia/jornais-investem-em-assinaturas-digitais-e-captam-novos-leitores>.

[14] SILVA, N. M. R. da. Fake news: a revitalização do jornal e os efeitos fact-checking e crosscheck no noticiário digital. *Temática*, v. 13, n. 8, 2017.

[15] AOSFATOS. *Nosso Método*. 2015. Disponível em <https://aosfatos.org/nosso-m%C3%A9todo/> Acesso em: 29 de agosto de 2018.

[16] GHOSH, S. *Can algorithms really tackle the fake news fiasco?* 2017. Disponível em <http://www.computerworld.in/feature/can-algorithms-really-tackle-fake-news-fiasco> Acesso em: 30 de agosto de 2018.

[17] CORNELL. *Pagerank – A mechanism to combat fake news?* 2017. Disponível em <https://blogs.cornell.edu/info2040/2017/10/18/pagerank-a-mechanism-to-combat-fake-news/> Acesso em: 30 de agosto de 2018.

[18] SHU, K. et al. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, ACM, v. 19, n. 1, p. 22–36, 2017.

[19] MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas Inteligentes-Fundamentos e Aplicações*, v. 1, n. 1, p. 32, 2003.

[20] MARTINS, C. B. et al. Introdução à sumarização automática. *Relatório Técnico RT-DC*, v. 2, n. 1, p. 35, 2001.

[21] GOLDBERG, Y.; LEVY, O. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[22] JR, E. C. T.; LIM, Z. W.; LING, R. Defining "fake news" a typology of scholarly definitions. *Digital Journalism*, Taylor & Francis, v. 6, n. 2, p. 137–153, 2018.

[23] APRÁ, A. *Levantamento feito com dados da USP embasa lista dos 10 maiores sites de "falsas notícias" no Brasil*. 2017. Disponível em <https://www.issoenoticia.com.br/artigo/projeto-da-usp-lista-10-maiores-sites-de-falsas-noticias-no-brasil> Acesso em: 29 de agosto de 2018.

[24] VICTOR, F. *Notícias falsas existem desde o século 6, afirma historiador Robert Darnton*. 2017. Disponível em <https://www1.folha.uol.com.br/ilustrissima/2017/02/1859726-noticias-falsas-existem-desde-o-seculo-6-afirma-historiador-robert-darnton.shtml> Acesso em: 30 de agosto de 2018.

[25] AVENDAÑO, F. B. T. C. *Fake News: a guerra informativa que já contamina as eleições no Brasil*. 2018. Disponível em <https://brasil.elpais.com/brasil/2018/02/09/politica/1518209427_170599.html>.

[26] SHAO, C. et al. The spread of fake news by social bots. *arXiv preprint arXiv:1707.07592*, arXiv, p. 96–104, 2017.

[27] MARKINES, B.; CATTUTO, C.; MENCZER, F. Social spam detection. In: ACM. *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*. [S.l.], 2009. p. 41–48.

[28] BAKIR, V.; MCSTAY, A. Fake news and the economy of emotions: Problems, causes, solutions. *Digital Journalism*, Taylor & Francis, v. 6, n. 2, p. 154–175, 2018.

[29] AGGARWAL, C. C.; ZHAI, C. *Mining text data*. [S.l.]: Springer Science & Business Media, 2012.

[30] FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine*, v. 17, n. 3, p. 37, 1996.

[31] NALLAPATI, R. et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. In: *CoNLL*. [S.l.: s.n.], 2016.

[32] HOVY, E.; LIN, C.-Y. Automated text summarization and the summarist system. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998*. [S.l.], 1998. p. 197–214.

[33] SRIVIDHYA, V.; ANITHA, R. Evaluating preprocessing techniques in text categorization. *International journal of computer science and application*, v. 47, n. 11, p. 49–51, 2010.

[34] MARSLAND, S. *Machine learning: an algorithmic perspective*. [S.l.]: Chapman and Hall/CRC, 2011.

[35] MURPHY, K. P. *Machine learning: a probabilistic perspective*. Cambridge, MA: [s.n.], 2012. 1–10 p.

[36] MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of machine learning*. [S.l.]: MIT press, 2012. 1 p.

[37] BISHOP, C. M. *Pattern Recognition and Machine Learning*. [S.l.]: Springer, 2006. 1–4 p.

[38] BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

[39] KINGSFORD, C.; SALZBERG, S. L. What are decision trees? *Nature biotechnology*, Nature Publishing Group, v. 26, n. 9, p. 1011, 2008.

[40] SILVA, L. Uma aplicação de árvores de decisão, redes neurais e knn para a identificação de modelos arma não sazonais e sazonais. *Disponível em <https://www.maxwell.vrac.puc-rio.br/7587/7587_4.PDF>*, 2005.

[41] CAMPBELL, W. M. et al. Support vector machines for speaker and language recognition. *Computer Speech & Language*, Elsevier, v. 20, n. 2-3, p. 210–229, 2006.

[42] JOACHIMS, T. Text categorization with support vector machines: Learning with many relevant features. In: SPRINGER. *European conference on machine learning*. [S.l.], 1998. p. 137–142.

[43] HAYKIN, S. S. et al. *Neural networks and learning machines*. [S.l.]: Pearson Upper Saddle River, 2009. v. 3.

[44] KOVÁCS, Z. L. *Redes neurais artificiais*. [S.l.]: Editora Livraria da Fisica, 2002.

[45] BRAGA, A. d. P.; CARVALHO, A.; LUDERMIR, T. B. *Redes neurais artificiais: teoria e aplicações*. [S.l.]: Livros Técnicos e Científicos Rio de Janeiro, 2000.

[46] DENG, L.; YU, D. *Deep Learning: Methods and Applications*. 2014. Disponível em <https://www.microsoft.com/en-us/research/publication/deep-learning-methods-and-applications/>.

[47] LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015.

[48] SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural networks*, Elsevier, v. 61, p. 85–117, 2015.

[49] TECHLABS, M. *Top 8 Deep Learning Frameworks*. 2018. Disponível em <https://www.marutitech.com/top-8-deep-learning-frameworks>.

[50] BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, v. 5, n. 2, p. 157–166, 1994.

[51] HORN, R. A. The hadamard product. In: *Proc. Symp. Appl. Math.* [S.l.: s.n.], 1990. v. 40, p. 87–169.

[52] PEDREGOSA, F. et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, v. 12, n. Oct, p. 2825–2830, 2011.

[53] REHUREK, R.; SOJKA, P. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, v. 3, n. 2, 2011.

[54] ABADI, M. et al. Tensorflow: A system for large-scale machine learning. In: *OSDI*. [S.l.: s.n.], 2016. v. 16, p. 265–283.

[55] CHOLLET, F. et al. Keras: The python deep learning library. *Astrophysics Source Code Library*, 2018.