UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS CRATEÚS

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FÁBIO JOSÉ DOS SANTOS SOUSA

TRANSFERÊNCIA DE CONHECIMENTO PARA DETECÇÃO AUTOMÁTICA DE FAKE NEWS COM APRENDIZAGEM PROFUNDA

CRATEÚS

2022

FÁBIO JOSÉ DOS SANTOS SOUSA

TRANSFERÊNCIA DE CONHECIMENTO PARA DETECÇÃO AUTOMÁTICA DE FAKE NEWS COM APRENDIZAGEM PROFUNDA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do CAMPUS CRATEÚS da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Me. Lívio Antônio Melo Freire

CRATEÚS

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S696t Sousa, Fábio José dos Santos.
Transferência de conhecimento para detecção automática de fake news com aprendizagem profunda / Fábio José dos Santos Sousa. – 2022.
74 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Crateús, Curso de Ciência da Computação, Crateús, 2022.
Orientação: Prof. Me. Lívio Antônio Melo Freire.

1. Fake News. 2. Machine Learning. 3. Deep Learning. 4. Big Data. 5. Transferência de conhecimento. I. Título.

CDD 004

FÁBIO JOSÉ DOS SANTOS SOUSA

TRANSFERÊNCIA DE CONHECIMENTO PARA DETECÇÃO AUTOMÁTICA DE FAKE NEWS COM APRENDIZAGEM PROFUNDA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do CAMPUS CRATEÚS da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em:

BANCA EXAMINADORA

Prof. Me. Lívio Antônio Melo Freire (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Ma. Cecília Silvestre Carvalho
Universidade de Fortaleza (Unifor)

Prof. Dr. José Wellington Franco da Silva
Universidade Federal do Ceará (UFC)

"O sonho é que leva a gente para frente. Se a gente for seguir a razão, fica aquietado, acomodado."

(Ariano Suassuna)

RESUMO

O uso de redes sociais para a comunicação e troca não medida de informação tem como consequência a disseminação de *fake news*. Por ter capacidade de influenciar e manipular o curso da realidade, o compartilhamento de *fake news* precisa ser coibido. Considerando o volume massivo de infamações que circulam nas redes sociais e sua velocidade de propagação, a revisão por profissionais capacitados, para a verificar a ocorrência de conteúdo não confiável, é inviável. Nesse sentido, ferramentas computacionais capazes de detectar automaticamente a ocorrência de *fake news* são necessárias. Neste trabalho, lidamos com o problema de classificar informação em formato de texto como falsa ou verdadeira, considerando unicamente a entrada. Assim, como abordagem, utilizando algoritmos de classificação de Aprendizagem de Máquina. Em particular, focamos em abordagens baseadas em Aprendizagem Profundo, pela capacidade de abstrair conceitos complexos e reconhecer padrões semânticos. Nesse tipo de abordagem, depende-se de conjuntos de dados rotulados, com volume suficiente para permitir o reconhecimento de padrões relativos ao problema. Como não se dispõe de conjuntos de dados volumoso na língua portuguesa, utilizamos técnicas de Transferência de Conhecimento para melhorar o desempenho dos algoritmos. Quanto à metodologia, trata-se de uma pesquisa exploratória e com natureza quantitativa, na qual investigamos o desempenho, por meio de experimentos computacionais, de métodos estado da arte para resolver o problema. No que diz respeito aos dados, será utilizado a base *Fake.br* para treinar e avaliar os modelos. .

Palavras-chave: *fake news*. Aprendizado Máquina. Aprendizado Profundo. Big Data, Transferência de conhecimento.

ABSTRACT

The use of social networks for communication and non-measured exchange of information results in the spread of *fake news*. Because it has the capacity to influence and manipulate the course of reality, the sharing of *fake news* needs to be restrained. Considering the massive volume of information circulating on social networks and its speed of propagation, the review by trained professionals, to verify the occurrence of unreliable content, is not viable. In this sense, computational tools capable of automatically detecting the occurrence of *fake news* are necessary. In this work, we deal with the problem of classifying information in text format as false or true, considering only the input. So, as an approach, using Machine Learning classification algorithms. In particular, we focus on approaches based on Deep Learning, for the ability to abstract complex concepts and recognize semantic patterns. In this type of approach, it relies on labeled data sets, with sufficient volume to allow the recognition of patterns related to the problem. As there are no large data sets available in the Portuguese language. We use Knowledge Transfer techniques to improve the performance of the algorithms. As for the methodology, it is an exploratory and quantitative research, in which we investigate the performance, through computational experiments, of state-of-the-art methods to solve the problem. Regarding the data, the base *Fake.br* will be used.

Keywords: Fake News. Machine Learning. Deep Learning. Big Data. Transfer learning

LISTA DE FIGURAS

Figura 1 – Bag-of-Words 19
Figura 2 – Exemplo de classificação de notícias baseado em tags 21
Figura 3 – Transferência de aprendizado 24
Figura 4 – Hiperplanos de representação 27
Figura 5 – Função de ativação 28
Figura 6 – Neurônio artificial e Uma rede neural simples 30
Figura 7 – Arquitetura de Rede Neural Recorrente Simples 31
Figura 8 – Arquitetura de Rede Neural Recorrente Bidirecional 31
Figura 9 – Transformers. 32
Figura 10 – Os dois tipos de atenção. 34
Figura 11 – fluxo dos passos da proposta 46
Figura 12 – Frequência das palavras 49
Figura 13 – Exemplo de tokenização 50
Figura 14 – Modelo LSTM. 54
Figura 15 – Modelo BI-LSTM. 55
Figura 16 – Matriz de confusão dos dados de teste 59
Figura 17 – Matriz de confusão dos dados de teste 61
Figura 18 – Matriz de confusão dos dados de teste 62
Figura 19 – Acurácia e função de perda do modelo em relação as épocas de treinamento, em azul valores para o conjunto de treinamento e em laranja para o conjunto de validação, em vermelho a função de perda da validação e em verde a de treino 63
Figura 20 – Matriz de confusão dos dados de teste 64
Figura 21 – Acurácia e função de perda do modelo em relação as épocas de treinamento, em azul valores para o conjunto de treinamento e em laranja para o conjunto de validação, em vermelho a função de perda da validação e em verde a de treino 64
Figura 22 – Matriz de confusão dos dados de teste 65

Figura 23 – Acurácia e função de perda, em relação a validação cruzada no treinamento, em azul a acurácia de treino e laranja a acurácia de validação, e as retas de baixo são referente a função de perda, onde a reta verde é *loss* de treinamento, enquanto que vermelho é a *loss* validação 66
Figura 24 – Matriz de confusão dos dados de teste 67
Figura 25 – Desempenho dos modelos no cenário das *fake news*, aqui temos a comparação dos modelos em relação as métricas de avaliação 67
Figura 26 – Desempenho dos modelos no cenário das notícias reais, aqui temos a comparação dos modelos em relação as métricas de avaliação 68

LISTA DE TABELAS

Tabela 1 – Exemplos de notícias manipuladas 17
Tabela 2 – Trabalhos relacionados e suas base de dados 44
Tabela 3 – Métricas dos trabalhos relacionados 44
Tabela 4 – Categorias do corpus *Fake.br* 48
Tabela 5 – Treinamento do modelo Naive Bayes com cros-validation e as métricas de avaliação 58
Tabela 6 – Métricas de avaliação para o teste de validação do Naive Bayes 59
Tabela 7 – Treinamento do modelo Logistic Regression com cros-validation e as métricas de avaliação 60
Tabela 8 – Métricas de avaliação para os dados de teste no Logistic Regression 60
Tabela 9 – Treinamento do modelo Support Vector Machine Classifer com cros-validation e as métricas de avaliação 61
Tabela 10 – Métricas de avaliação para os dados de teste no Support Vector Machine Classifier 62
Tabela 11 – Métricas de avaliação para os dados de teste na LSTM simples 63
Tabela 12 – Métricas de avaliação para os dados de teste na LSTM bidirecional 65
Tabela 13 – Métricas de avaliação para os dados de teste no BERT 66

LISTA DE ABREVIATURAS E SIGLAS

[CLS]	*token* inserido no começo da sentença
[SEP]	*token* inserido no final da sentença
AM	Aprendizagem de Máquina
AP	Aprendizagem Profunda
BERT	*Bidirectional Encoder Representations from Transformers*
BI-LSTM	*Bidirectional long-short term memory*
CNN	*convolutional neural network*
IA	Inteligência Artificial
LSTM	*Long Short Term Memory*
MLM	*Masked LM*
NSP	*Next Sentence Prediction*
PLN	Processamento da Linguagem Natural
RN	Redes Neurais Artificiais
SVC	*Support Vector Classifier*
SVM	*Support vector machine*
TC	Transferência de Conhecimento
TF	Frequência de termo
TF-IDF	*term frequency-inverse document frequency*

SUMÁRIO

1 INTRODUÇÃO 13
1.1 Contextualização 13
1.2 Justificativa 15
1.3 Objetivos 16
1.3.0.1 Objetivos Específicos *16*
2 FUNDAMENTAÇÃO TEÓRICA 17
2.1 Características das *fake news* 17
2.2 Definição de classificação de texto 19
2.3 Aprendizado de Máquina 20
2.4 Aprendizado Profundo 21
2.5 Transferência de Conhecimento 23
2.6 Algoritmos 26
2.6.1 Naive Bayes 26
2.6.2 Support Vector Machine 26
2.6.3 Logistic Regression 27
2.6.4 Redes Neurais 28
2.6.5 Redes Neurais Recorrentes 30
2.6.6 Transformers 31
2.6.7 BERT 35
3 TRABALHOS RELACIONADOS 38
4 METODOLOGIA 45
4.1 Visão geral 45
4.1.1 Fluxograma da proposta 46
4.2 Métodos e Modelos 46
4.2.1 Conjunto de Dados 47
4.2.2 Escolha dos modelos 48
4.2.3 Métodos 48
4.2.4 Modelagem 51
4.2.4.1 Definição da modelagem dos dados *51*
4.2.4.2 Multinomial Naive Bayes *51*

4.2.4.3 Logistic Regresssion 52
4.2.4.4 Support vector machine 52
4.2.4.5 LSTM(Long Short Term Memory) 53
4.2.4.6 BI-LSTM(Bidirectional long-short term memory) 54
4.2.4.7 BERT 55
4.3 Treinamento e Validação 56
4.4 Considerações finais do capítulo 56
5 RESULTADOS 57
5.1 Métricas 57
5.2 Resultados dos modelos de aprendizado de máquina 58
5.2.1 Naive Bayes 58
5.2.2 Avaliação do Naive Bayes 59
5.2.3 Logistic Regression 59
5.2.4 Avaliação da Logistic Regression 60
5.2.5 Support Vector Machine 61
5.2.6 Avaliação do Support Vector Machine 61
5.3 Resultados dos modelos de aprendizado profundo 62
5.3.1 LSTM 62
5.3.2 Avaliação da LSTM 63
5.3.3 BI-LSTM 64
5.3.4 Avaliação da BI-LSTM 65
5.3.5 BERT 65
5.3.6 Avaliação do BERT. 66
5.4 Comparação dos experimentos 67
6 CONCLUSÕES E TRABALHOS FUTUROS 69
REFERÊNCIAS 71

1 INTRODUÇÃO

1.1 Contextualização

Desde o princípio, a humanidade exerce a comunicação mediante a troca de informações através de interação social. Entretanto, o ato de comunicação está sujeito a falhas, oriundas da falta de percepção do mundo à sua volta ou da intenção de alterar a visão que o próximo tem sobre os fatos do mundo.

O conceito de *fake news* refere-se à alteração deliberada da percepção do próximo por meio de transmissão de notícias com informações manipuladas. Essas alterações têm como propósito convencer grupos sobre fatos dissociados da realidade, para atingir objetivos de convencimento em cenários políticos, religiosos, comerciais, etc. O convencimento se dá pelo envio massivo de informações manipuladas, que alimentam esses grupos e os convencem, por meio da repetição e apelo emocional, a negar a realidade e crer naquilo que o agente propagador deseja.

O termo *fake news* ganhou força em 2016, na campanha eleitoral para presidente dos EUA, em que disseminação de informação falsa exerceu influência no desfecho da eleição (ALLCOTT; GENTZKOW, 2017). Nesse caso, a manipulação da informação teve como propósito arregimentar um grupo político sólido e crescente, que adotou o candidato Donald Trump como única solução para as crises criadas pelas *fake news* e ignorou os fatos divulgados pela imprensa oficial que colocavam em dúvida a idoneidade do candidato. Para atingir esse fim, a campanha contratou a empresa *Cambridge Analytica*, que analisou dados das redes sociais de cidadãos americanos para identificar o público propenso a aderir às informações manipuladas.

O mecanismo de utilizar a desinformação para atingir objetivos específicos, entretanto, já existe há bastante tempo. De acordo com (FOLHA DE S.PAULO, 2017), informações manipuladas vêm sendo espalhadas desde o século VI. Um caso notável é do historiador bizantino Procópio, famoso por escrever a história do império de Justiniano. Ele também escreveu o texto secreto chamado “Anekdota”, repleto de informações falsas, com propósito de arruinar completamente a reputação do imperador Justiniano e de outras personalidades. Os objetivos eram também políticos, similar ao que aconteceu na campanha eleitoral americana (POLITIZE, 2017).

O usuário propenso a consumir *fake news* costuma disseminar a notícia recebida com seus amigos e familiares, nas redes sociais e em aplicativos de mensagens, fazendo com que

a informação circule e aumente a adesão ao grupo. Para combater a crescente desinformação, inclusive no contexto das próprias redes sociais, vários sistemas de checagem de fatos foram criados para confirmar a veracidade dos fatos descritos numa notícia. Esses sistemas são cruciais para o combate de *fake news* em um mundo de política da pós-verdade (MIHALCEA; STRAPPARAVA, 2009). O conceito de pós-verdade refere-se a apelos sentimentais, em que as crenças pessoais são mais importantes que a veracidade dos fatos. Portanto, as *fake news* carregam elementos personalizados, esquematizados para convencer um público específico.

A preocupação com problema das *fake news* e suas consequências vem sendo objeto de pesquisa em diversas áreas do conhecimento, com estudos nas áreas de ciências humanas, sociais e exatas. Em particular, este estudo foca na aplicação de técnicas baseadas em dados para detectar automaticamente se uma notícia se trata de *fake news* ou não. Assim, este estudo insere-se no cenário de Big Data, no qual se dispõe de quantidade massiva de dados, que podem ser utilizados para auxiliar em processos de tomada de decisão (OTTONICAR *et al.*, 2019). A utilização de tecnologias de Big Data permite, por exemplo, segmentar o público propenso a se engajar em campanhas para promoção de candidatos ou marcas, gerando concorrência desleal. A base dessas tecnologias são os dados de usuários, que registram interesses pessoais, permitindo a segmentação de grupos específicos. A partir da caracterização desses grupos, empresas propagam conteúdo personalizado, seja para promover itens de interesse ou desacreditar a concorrência. Nessa perspectiva, observa-se que as análises realizadas no âmbito de Big Data podem ser usadas para o bem ou para o mal.

Com a crescente disponibilidade de bases de dados contendo notícias marcadas como *fake news* ou não, permite-se a utilização de técnicas automáticas para reduzir o fluxo das *fake news*. Essas técnicas procuram descobrir padrões que caracterizam notícias falsas, como erros ortográficos, número de referências para fontes, tamanho da mensagem, termos específicos, etc. A utilização de métodos baseados em dados para automatizar a tarefa de detecção de *fake news* vem aliviando o trabalho antes restrito a pessoas, que verificam manualmente os fatos contidos numa notícia para decidir sobre a veracidade da informação.

Existem trabalhos correlatos no âmbito da detecção automática de *fake news*, que diferem sobre a modelagem do problema e a técnica utilizada (YANG *et al.*, 2018; PÉREZ-ROSAS *et al.*, 2017). Como trata-se de problema de difícil generalização, tendo em vista a dificuldade para interpretar automaticamente a linguagem natural para tomar decisões, há limitações em cada uma dessas abordagens. Mesmo quando se dispõe de conjuntos de dados robustos, os mo-

delos acabam sendo enviesados para memorizar padrões, dificultando a generalização. Técnicas de Aprendizagem Profunda (AP) buscam amenizar esse problema, com modelos voltados a interpretar o texto, em vez de focar apenas em termos específicos, como fazem os modelos de Aprendizagem de Máquina (AM) clássicos. Entretanto, para funcionar bem, os modelos de AP dependem da disponibilidade de um conjunto representativo de dados sobre a tarefa alvo, que nem sempre está disponível.

Recentemente, técnicas de Transferência de Conhecimento (TC) vem sendo utilizadas para melhorar a generalização de modelos de AM. Nessas abordagens, parte-se de um modelo robusto capaz de compreender a linguagem natural, ajustado sobre um conjunto de dados representativo e independente da tarefa alvo, seguida da etapa de ajuste fino, que fornece a esse modelo a habilidade de tomar decisões sobre a tarefa alvo.

Em (YANG *et al.*, 2018), utiliza-se técnicas de AP e TC para detecção automática de notícias falsas. Esse trabalho compara técnicas diferentes de AM para mostrar o ganho do emprego de TC a modelos de AP para a tarefa de detecção de *fake news*. Em (PÉREZ-ROSAS *et al.*, 2017), emprega-se um modelo unificado para detectar *fake news*, utilizando dados de texto e imagem. Esse modelo unificado combina dois problemas em um só. O terceiro usa abordagem linguísticas para automatizar a detecção de notícias falsas. Esse trabalho sugere que conteúdo falso e conteúdo genuíno deve ser examinado a fundo, nos níveis sintático e semântico.

Os trabalhos mencionados fornecem uma visão sobre técnicas estado da arte para a detecção automática de *fake news*. Assim, conclui-se que trabalhos que usam AP se saem melhor nessa tarefa. Outro ponto de destaque é que técnicas de TC podem ajudar mais ainda na generalização, com reuso de tarefas já aprendidas visando à compreensão da linguagem natural, permitindo que os modelos sejam mais precisos na detecção automática de *fake news*.

1.2 Justificativa

A detecção automática de *fake news* é uma tarefa necessária na era de pós-verdade, considerando a disseminação em massa de informações com fatos falsos para manipular a opinião de multidões. Esse problema pode ser modelado como uma tarefa de classificação de texto, subtarefa de AM, desde que se disponha de conjunto de dados contendo notícias marcadas como falsas ou verdadeiras. Na área de AP, existem modelos propostos voltados para lidar com entradas no formato textual (MITTAL, 2019). Esses modelos AP funcionam bem quando um conjunto de dados representativo está disponível para a tarefa de destino. No entanto, quando os

dados são limitados, o modelo pode perder alguma eficiência e, eventualmente, sobreajustando ao conjunto de treinamento, concentrando-se em termos-chave, perdendo assim a capacidade de interpretação. Para solucionar esse problema, é utilizado o método TC, que permite ao modelo AP utilizar uma grande quantidade de dados para pré-treinamento para outra tarefa, ganhando a capacidade de interpretar texto em linguagem natural. Nesse contexto, técnicas de TC podem melhorar na generalização da detecção de *fake news*, como no trabalho de (CRUZ *et al.*, 2019).

Este trabalho contribui para a detecção automática de notícias falsas em português usando o modelo AP com TC. Os trabalhos encontrados na literatura, que aplicam técnicas estado da arte para o problema, analisam o desempenho dessa tarefa para o idioma inglês. Em (MONTEIRO *et al.*, 2018), aborda-se a classificação de *fake news* para língua portuguesa utilizando técnicas clássicas de AM, além de contribuir com o corpus de *fake news Fake.br*, que contém noticias rotuladas como verdadeiras ou falsas. Nessa perspectiva, este trabalho se justifica ao comparar modelos clássicos de AM com técnicas avançadas de AP com TC sobre o conjunto de dados *Fake.br*.

1.3 Objetivos

Investigar se abordagens de **transferência de conhecimento** melhoram na performance das soluções baseadas em **aprendizagem profunda** para o problema de **detecção automática de *fake news***.

1.3.0.1 Objetivos Específicos

1. Avaliar se à **integração de modelos neurais pré-treinados** melhora na **classificação automática de *fake news*** em **textos**.
2. Avaliar se a **adaptação de domínios em modelos de linguagem pré-treinados** melhora a **classificação automática de *fake news*** em **textos**.
3. Identificar a melhor abordagem baseada em **modelos pré-treinados**.
4. Avaliar os melhores **modelos pré-treinados** com pequenos ajustes em seus **atributos** como mudar o tamanho da sequencia, curva de aprendizado ou nas épocas com intuito de verificar se afeta de forma positiva ou negativa.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, abordaremos conceitos gerais para o estudo das *fake news*. Na primeira seção 2.2, é apresentada a definição de classificação de texto. Na seção 2.3, são definidos os conceitos de AM em classificação de texto. Na seção 2.3, são apresentados os conceitos de classificação de texto em AP. Na seção 2.5, são abordadas as definições e notações sobre TC. E por fim na seção 2.6 são introduzidos modelos para a classificação automática de texto.

2.1 Características das *fake news*

O termo *fake news* surgiu na mídia tradicional para se referir a informações enganosas divulgadas por fontes não confiáveis. Eles são apresentados em formato jornalístico como se tivessem informações factuais válidas, mas apresentam uma realidade distorcida e sensacionalista projetada para enganar e manipular a multidão.

O conteúdo manipulado, muitas vezes, parte de uma notícia inicialmente verdadeira, alterada para enganar e manipular os leitores, especialmente aqueles propícios a adotar a informação como verdade sem verificar os fatos. O conceito se estendeu para postagens em redes sociais e aplicativos de troca de mensagens, utilizados para disseminar com mais agilidade as informações enganosas. Segundo (JR *et al.*, 2018), as *fake news* são criadas com características para serem compartilhadas nas redes sociais. Exemplo de conteúdo manipulado é mostrado na Tabela 1.

Tabela 1 – Exemplos de notícias manipuladas

Falso	Verdadeiro
Michel Temer propõe fim do carnaval por 20 anos, "PEC dos gastos". Michel Temer afirmou que não deve haver gastos com aparatos supérfluos sem pensar primeiramente na educação do Brasil. A medida pretende calcelar o carnaval de 2018.	Michel Temer não quer o fim do Carnaval por 20 anos. Notícias falsas misturam proximidade dos festejos, crise econômica e medidas impopulares do governo do peemedebista.
Acabou a mordomia ! Ingresso mais barato pra mulher é ilegal. Baladas que davam meia entrada para mulher, ou até mesmo gratuidade, esto na ilegalidade agora. Acabou o preconceito com os homens nas casas de show de todo o Brasil.	Ingresso feminino barato como marketing 'não inferioriza mulher', diz juíza do DF. Afirmação consta em decisão sobre preços diferentes para homens e mulheres em festa no Lago Paraná. 'Prática permite que mulher possa optar por participar de tais eventos sociais', diz texto.

Fonte: (MONTEIRO *et al.*, 2018)(Fake.br)

Conforme mostrado na Tabela 1, o conteúdo das notícias pode ser facilmente manipulado, seja sobre política, economia, religião, ciência e tecnologia, sociedade, notícias diárias, TV

e celebridades. Considerando essas categorias, o problema de detecção de *fake news* apresenta dificuldades em detectar notícias falsas, tendo em vista a diversidade do vocabulário e as variedades linguísticas de uma linguagem natural. Um dos primeiros passos para detectar *fake news* é analisar a posição do corpo de notícias e da entidade que está descrevendo (THOTA *et al.*, 2018). Esta tarefa pode ser considerada um problema de Inteligência Artificial (IA), mais precisamente na subárea de Processamento da Linguagem Natural (PLN). O PLN corresponde à forma dos computadores analisarem, compreenderem e derivarem o significado da linguagem humana de forma inteligente e útil. Usando o PLN, os desenvolvedores podem organizar e estruturar o conhecimento para realizar tarefas como Sumarização Automática, Tradução, Análise de Sentimento, etc. (WARNER, 1987). A tarefa de detectar *fake news* pode ser descrita como um procedimento de classificação de texto, para prever automaticamente se uma entrada de notícias é uma *fake news* ou não.

A ideia de classificar *fake news* consiste em aprender vários níveis de abstração sobre a linguagem natural, para ser capaz de identificar combinações de características que indicam informação falsa ou não. Em abordagens de AM, parte-se de um conjunto de características extraídas sobre os documentos de entrada (conjunto de treinamento), dos quais já se sabe a classe (*fake news* ou não). Esse conjunto de treinamento servirá para que o algoritmo de aprendizado de máquina aprenda diferentes associações sobre as características, gerando um modelo capaz mapear características para uma das classes. Assim, um documento que não faz parte do conjunto de treinamento poderá ser classificado apenas com base na extração se suas características. Dessa forma, a primeira etapa na criação de um classificador é a modelagem e extração de características. O método *Bag-of-Words* é amplamente utilizado em AM para extrair características de textos. Nessa abordagem usa um vetor que representa a frequência de uma palavra em um dicionário de palavras predefinido. Por exemplo, se quiséssemos vetorizar a frase “isto é incrível", obteríamos o seguinte vetor de saída (1, 1,0,0,1,0,0) como representa na Figura 1. Claro, se você considerar o dicionário pré-existente { Isto, é, não, o, incrível, ruim, basquete } (MOKEYLEARN, 2019). Depois disso, o modelo seria alimentado com este com os vetores e as categorias produzindo uma classificação de texto. Depois disso, o modelo inicia sua fase de teste, para obter previsões de dados não rotulados, porém, existem desvantagens porque o modelo treinado com esses dados específicos serão geralmente usados para treinar outro *dataset* com características semelhantes. Para suprir essa perda de dados, surgiu o aprendizado profundo.

Figura 1 – Bag-of-Words

	Document	abreu	agricultura	cargo	citado	colocar	convida	cupula	daniel	deputado	dilma	expulsao	katia
0	corpus[0]	2	0	0	0	1	0	1	0	0	0	2	2
1	corpus[1]	2	0	0	0	0	1	0	0	0	2	0	2
2	corpus[2]	1	1	1	1	0	0	0	1	1	0	0	1

Fonte: Autor

No Aprendizado Profundo os modelos procuram representar esses recursos de uma forma mais abstrata, ou seja, procuram ser mais semânticos em relação à classificação do texto. Algoritmos de aprendizado profundo tentam aprender a representação desses recursos abstratos e classificá-los. Eles não apenas têm a habilidade de descobrir padrões ocultos nos dados, mas também são muito mais transferíveis de um *dataset* para outro. Estes modelos de Aprendizado Profundo, hoje são as principais formas nas tarefas de classificação conseguindo um excelente desempenho e um excelente aproveitamento dos dados. Isso é alcançado por meio de níveis de conceito que permitem que os modelos definam conceitos complexos a partir de conceitos minimalistas. A forma como o aprendizado em profundidade utiliza suas funções não lineares em conceitos complexos para entender problemas vai além de qualquer mera representação do aprendizado de máquina, podendo mesmo inferir dados não rotulados a partir de dados já utilizados (MOKEYLEARN, 2019). Um exemplo de modelo de representação de classificação de texto são Redes Neurais Recorrentes, transformadores, *Bidirectional Encoder Representations from Transformers* (BERT). Esses são modelos poderosos para classificação de texto. Nas próximas seções, esses conceitos serão discutidos com mais profundidade.

2.2 Definição de classificação de texto

Na classificação do texto, é fornecida uma descrição $d \in \mathbb{X}$ de um documento, onde $\mathbb{X}$ é o espaço do documento; e um conjunto fixo de classes $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$. Classes, mas também são chamados de categorias ou rótulos. Normalmente, o espaço do documento $\mathbb{X}$ é algum tipo de espaço de alta dimensão, e as classes são definidas por humanos para as necessidades de um aplicativo, por exemplo, a classificação de *spam*. Aonde é recebido um conjunto de dados $\mathbb{D}$ de treinamento marcado no documento (d,c), onde $(d,c) \in \mathbb{X}$ x $\mathbb{C}$. Usando o método de aprendizagem deseja-se aprender a classificar um documento para que o classificador ou função de classificação y que visa mapear tags do documento para classes(CERI *et al.*, 2013):

$$F : \mathbb{X} \rightarrow \mathbb{C} \tag{2.1}$$

Esse tipo de aprendizado pode ser chamado de aprendizado supervisionado, pois define rótulos para o documento, aonde denota o método de aprendizagem sendo $F(\mathbb{D}) = y$ que recebe como entrada um conjunto $\mathbb{D}$ de treinamento e retorna uma função de classificação aprendida y. A maioria dos modelos de aprendizagem de máquina e aprendizagem profundo usam esse método para classificar as classes dos classificadores. Na próxima seção será explicado os conceitos de Aprendizado de Máquina(CERI *et al.*, 2013).

2.3 Aprendizado de Máquina

O aprendizado de máquina é uma área da Inteligência Artificial que tem como foco a produção de algoritmos capazes de aprender a resolver problemas complexos de forma autônoma. Esses algoritmos são capazes de reconhecer e extrair padrões de um extenso volume de dados, através de modelos criados com o objetivo de tomar decisões e previsões mais próximas do exato, com base nos padrões descobertos (MARSLAND, 2015). O conjunto de dados utilizado para a aprendizagem consiste em várias características, sendo definidas como variáveis (MURPHY, 2012)

Aprendizado de máquina pode ser classificado em dois tipos: supervisionado e não supervisionado. O aprendizado supervisionado é um conjunto de exemplos rotulados com dados de treinamento e faz previsões para todos os pontos não vistos. Este é o cenário mais comum associado a problemas de classificação e regressão. Mais precisamente esse aprendizado faz treinamento do conjunto de dados rotulado e fornece a saída desejada (as categorias predefinidas). Durante a fase de teste, o algoritmo é alimentado com dados não observados e os classifica em categorias com base na fase de treinamento(MOHRI *et al.*, 2018).

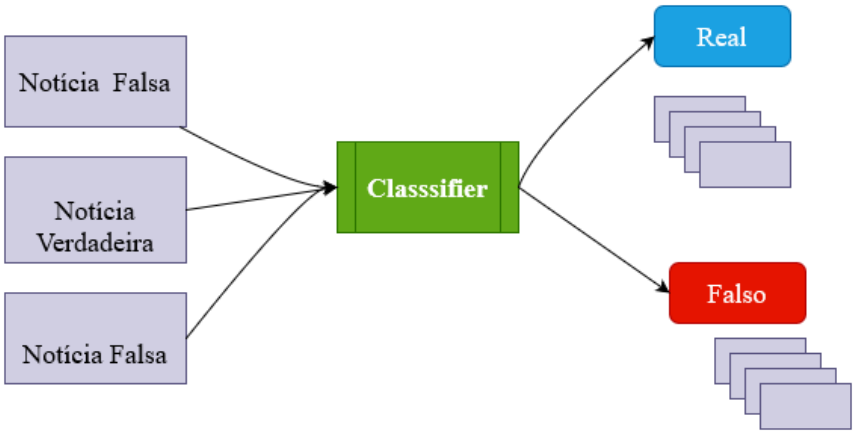
Segundo (MOHRI *et al.*, 2018) a filtragem de *spam* de *e-mails* é um exemplo de classificação supervisionada. O *e-mail* recebido é categorizado automaticamente com base em seu conteúdo. A detecção de linguagem, intenção, emoção e análise de sentimento são todas baseadas em sistemas supervisionados.

O aprendizado não supervisionado tem como objetivo classificar um conjunto com dados de treinamento não rotulados e faz previsões para todos os pontos não vistos. Como, em geral, nenhum exemplo rotulado está disponível nesse ambiente, pode ser difícil avaliar quantitativamente o desempenho do modelo. O agrupamento e a redução da dimensionalidade são exemplos de problemas de aprendizagem não supervisionados.(MOHRI *et al.*, 2018). Um exemplo de aprendizado não supervisionado é o agrupamento de documentos similares com base

no texto.

Conforme mencionado na sessão anterior, o Aprendizado de Máquina precisa vetorizar recursos para que esses modelos possam ser treinados. Uma vez treinados, eles estão prontos para fazer previsões. Além disso, *Bag-of-Words* também pode ser usado para transformar texto invisível em conjuntos de recursos que podem ser inseridos no modelo de classificação para obter previsões sobre as tags (por exemplo. Política, Economia, Religião, Ciência e Tecnologia, Sociedade e Notícias Diárias, TV e Celebridades). Para melhor compreensão observe a imagem abaixo; nela um modelo genérico de Aprendizagem de Máquina faz a classificação de texto em notícias. Na Figura 2 ilustra o exemplo da classificação de texto.

Figura 2 – Exemplo de classificação de notícias baseado em tags



Fonte: Autor

Os modelos de AM representam bem os dados e fazem previsões com muita precisão, no entanto, eles ainda precisam de alguma orientação. Se um algoritmo de AM retornar uma previsão imprecisa, um engenheiro terá que intervir e fazer ajustes. Com modelos de aprendizado profundo, os algoritmos podem determinar por si próprios se a suas previsões são precisas ou não. Na próxima sessão, será discutido mais a fundo sobre o Aprendizado Profundo.

2.4 Aprendizado Profundo

Aprendizado profundo(ou AP) é uma subárea de aprendizagem de máquina, que pode ser compreendida como programas que conseguem aprender com as suas experiências e podem extrair padrões complexos em diferentes níveis de abstração, usando várias camadas de processamento não lineares(ARAÚJO *et al.*, 2017).

Segundo (GOODFELLOW *et al.*, 2016), esses níveis de abstração permitem que programas de AP compreendam conceitos complexos a partir de recursos minimalistas. Esses recursos são extraídos através de um extrator de recurso que representa os dados brutos em valores, que podem ser uma representação interna ou um vetor de recursos sendo possível detectar, ou classificar padrões na entrada. Aprendizagem por representação é um conjunto de métodos que permite que uma máquina seja alimentada com dados brutos e descubra automaticamente as representações necessárias para a detecção ou classificação. Métodos de aprendizagem profunda são métodos de aprendizagem de representação com vários níveis de representação, obtidos pela composição de módulos simples, mas não lineares, em que cada um transforma a representação em um nível (começando com a entrada bruta) em uma representação em um nível superior, ligeiramente chamado nível abstrato. Com a composição de tais transformações, funções muito complexas podem ser aprendidas. Para tarefas de classificação, que com camadas superiores de representação amplificam aspectos da entrada importantes para a discriminação e suprimem variações irrelevantes(LECUN *et al.*, 2015). O aspecto principal é que com essas camadas de representação os modelos de aprendizagem profundo conseguem lidar com as limitações de Aprendizado de Máquina, e compreender estruturas complexas em dados de alta dimensão, portanto, aplicáveis a muitos domínios. A seguir serão abordadas essas limitações entre modelo com Aprendizado de Máquina e modelo com Aprendizado Profundo.

Há duas diferenças principais com as representações de modelo de espaço vetorial tradicional. A primeira diferença é que no modelo de espaço vetorial tradicional, temos uma representação extremamente esparsa, ou seja, cada objeto tem vários valores diferentes de zero, em média, enquanto o resto dos valores são zero (digamos, centenas diferentes de zero vs milhares de zeros). Ao contrário, para redes neurais, os valores são sempre números reais (geralmente no intervalo [0,1]), portanto, eles praticamente nunca se tornam zeros. Assim, cada objeto tem algum valor para qualquer característica. Nesse sentido, podemos comparar o aprendizado profundo com a análise semântica latente: não há tantas dimensões e essas dimensões têm valores diferentes de zero. A segunda diferença está relacionada à seleção de recursos. Até agora, foi considerado a representação de objetos usando as suas características e os valores dessas características, selecionados por pesquisadores. Com o aprendizado profundo, como, por exemplo, a arquitetura da rede neural, capaz de escolher automaticamente os pesos dos neurônios, equivalentes aos valores dos recursos. Como os neurônios representam diferentes níveis de generalização, a rede dar um bom desempenho(SIDOROV, 2019).

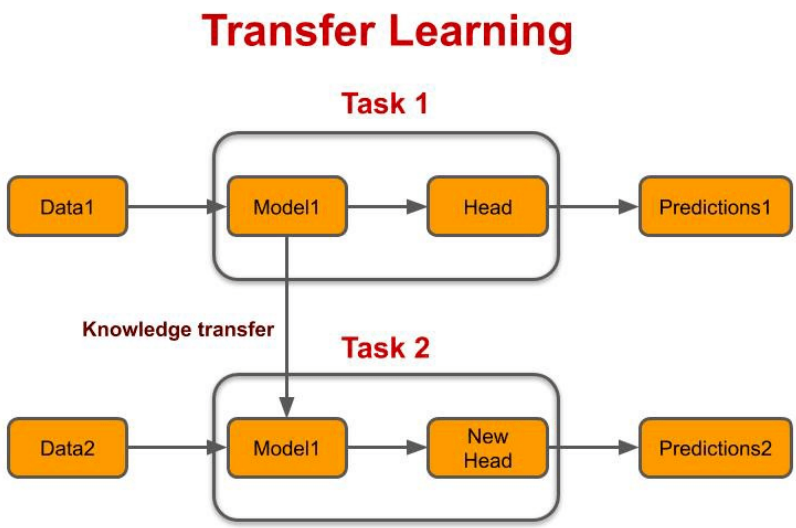
No entanto, as redes neurais voltadas para classificação de texto tem limitações como os dados. Essas limitações são: sequências muito complexas, relacionar sentenças complexas, a segunda palavra só pode entrar na rede depois que a primeira palavra já passou por ela, entre outras. Dependo da frase de entrada, a tarefa de classificação de texto não trata de alinhamento simples. Por isso, surgiu mecanismo de autoatenção (ou *self-attention* que foca em trechos específicos das frases de entrada. Também é importante ressalta que são todos modelos que usam Aprendizado Profundo. A proposta deste trabalho usará métodos de Aprendizado profundo que se encaixam nos modelos introduzidos, que serão abordados posteriormente nas próximas seções.

Além disso, Aprendizado Profundo proporcionou a esses modelos citados acima, a redução do tempo de treinamento evitando criar modelos do zero, além de tentar suprir a falta de grandes conjuntos de dados para o treinamento dos modelos. A solução foi reusar o conhecimento dos modelos previamente treinados com grandes volumes de dados. Esse reuso de dados pode ser chamado de transferência de conhecimento.

2.5 Transferência de Conhecimento

Transferência de Conhecimento é um método usado em aprendizagem profunda que faz o reuso de uma tarefa que já foi apreendido anteriormente em um modelo e a utiliza como ponto de partida para treinar modelo para uma segunda tarefa. Por exemplo, o conhecimento adquirido ao aprender a reconhecer carros pode ser aplicado para reconhecer caminhões. Abaixo a Figura 3 ilustra como é realizado a transferência de conhecimento.

Figura 3 – Transferência de aprendizado



Fonte: (MOLTZAU, 2019)

O objetivo dessa abordagem é extrair o conhecimento adquirido por um modelo a partir de uma ou mais tarefas de origem e aplicar em uma tarefa de destino. Uma razão para usar técnicas de transferência é que os domínios de origem e destino devem estar relacionados. Segundo (TAN *et al.*, 2015), essa relação pode ser concretizada na forma de instâncias(BICKEL *et al.*, 2009) ou características(SATPAL; SARAWAGI, 2007). Se não houver nenhuma relação, a transferência não vai funcionar.

Primeiramente serão apresentadas algumas notações e definições que foram apresentadas por (PAN; YANG, 2009) na transferência de conhecimento: um domínio D consiste em dois componentes: um espaço de características χ e uma distribuição de probabilidade marginal $P(X)$, em que $X = \{x_1 \ldots x_n\} \in \chi$. Em classificação de texto cada termo do documento é tido como uma característica binaria, então χ é o espaço em que quaisquer vetores de termos do documento, é x_i sendo o i-ésima o vetor de termos correspondente a algum documento e X é uma amostra. Normalmente, se dois domínios diferirem, então estes possuem espaços de distribuição probabilidade marginais diferentes.

As técnicas de transferência de conhecimento podem ser categorizadas. Segundo (PAN; YANG, 2009) as técnicas de transferência de aprendizado são: indutiva, transdutiva ou sem supervisão, com base nos possíveis cenários dados elas podem ser usadas.

Na Transferência de conhecimento indutiva, as tarefas de destino e origem diferem, mas os domínios são os mesmos. Sendo estes $T_s \neq T_t$ tendo dois casos.

1. Onde os espaços de rótulos são distintos ($Y_s \neq Y_t$). ou

2. A probabilidade condicional entre os domínios são $P(Y_s|X_s) \neq P(Y_t|X_t)$. sendo $f_s(.) \neq f_t(.)$

Na classificação de texto, o caso 1 refere-se ao cenário onde o domínio de origem possui uma classificação binária e o domínio de destino possui uma classificação multiclasse. No segundo caso é quando a classificação de texto é desbalanceada em um dos domínios. A abordagem que a transferência de conhecimento indutiva usa é similar à aprendizagem multitarefa, quando há dados nos dois domínios, entretanto, ela é restrita apenas ao domínio de destino. Um exemplo dessa abordagem em classificação de texto é no reconhecimento de dígitos e no reconhecimento de letras.

Na transferência de conhecimento transdutiva, as tarefas são as mesmas de origem e destino, já os domínios são diferentes $\chi_s \neq \chi_t$ ou $P(X_s) \neq P(X_t)$. Normalmente, não há dados rotulados disponíveis no domínio de destino, já no de origem, há bastante dados rotulados disponíveis. No domínio de origem os dados são divididos em grupos de domínios rotulados específicos e com base nesses subdomínios rotulados é feito um ajustado iterativamente para escolher os melhores hiperparametros em cada um deles, que serviram de estimativa para o domínio de destino não rotulado, por exemplo, na classificação de *fake news* podemos ter vários subdomínios específicos de *fake news*, como *fake news* política, religião. Nos dois subdomínios rotulados será escolhido o melhor *score label*, e será usado como um ponto no domínio de destino. Conhecidos como pseudos rótulos. Na fase de rotulagem desse pseudo-rotulo exigido que alguma instância do domínio de destino sejam conhecidas, isso auxiliarar no processo de escolha do melhor pseudo-rotulo. No exemplo de *fake news* o que os subdomínios têm em comum é apenas o fato de ser falso.

A transferência de aprendizado não supervisionado é análogo à transferência indutiva. No entanto, esse método centraliza no aprendizado não supervisionado no domínio de chegada, como agrupamento, redução de dimensionamento e estimação de densidade. Assim, não há rotulados nos domínios de origem e destino. Isto é, todo o procedimento é supervisionado durante a fase de pré-treino, mas não no decorrer do ajuste fino. Esta técnica é conhecida como auto-identificação, que com base nela o modelo aprenderá sozinho a rotular dados novos providos de um novo *dataset*, isso possibilita a modelagem de estimativas das probabilidades de entrada para esses novos dados. Um exemplo dessa abordagem, seria a aplicação do conhecimento adquirido para a redução de características de um problema de imagens do mundo real para imagens de documentos.

Nas próximas seções serão abordados conceitos que fundamentaram a hipótese da pesquisa. Iniciando pela seção 2.6 que aborda conceitos sobre os algoritmos que estão presente nesta pesquisa. Que vão de algoritmos tradicionais de aprendizado de máquina até algoritmos robustos de aprendizado profundo!Em cada subseção são abordados os princípios fundamentais de cada um.

2.6 Algoritmos

seção exclusiva para exemplificar os algoritmos que serão usado no desenvolvimento da pesquisa.

2.6.1 Naive Bayes

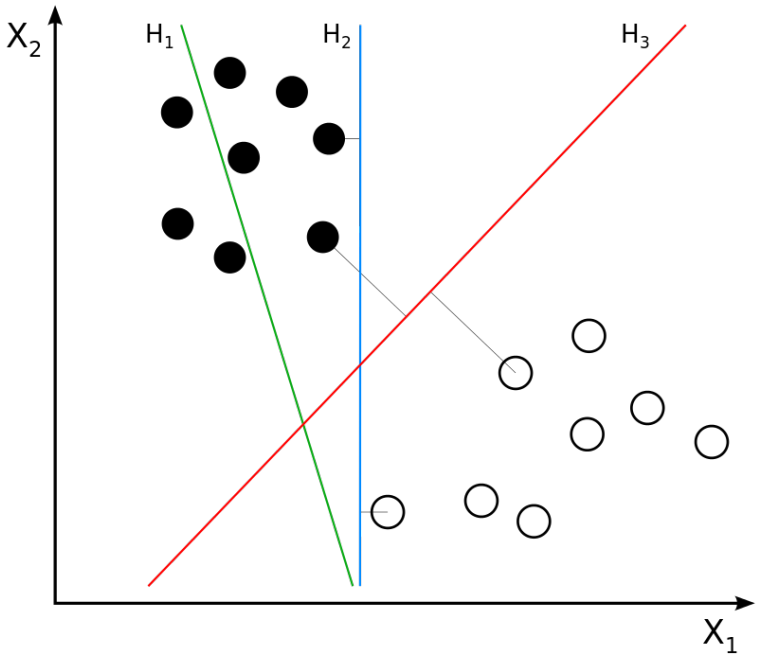
Naive Bayes é um algoritmo de modelagem preditiva simples, mas poderoso. Baseia-se em técnicas estatísticas de probabilidade condicional, baseadas no teorema de Thomas Bayes.(BAYES, 2022) Uma vez calculado, um modelo probabilístico pode ser usado para fazer previsões sobre novos dados usando o Teorema de Bayes.

Um evento ocorre dada a probabilidade de que outro evento tenha ocorrido: $P(B|A) = \frac{P(A \cap B)}{P(A)}$. O algoritmo assume que cada variável de entrada é independente, mas nem todos os problemas com esse método são relevantes, portanto, existem várias variantes do algoritmo para lidar com diferentes tarefas.

2.6.2 Support Vector Machine

Support Vector Machine (SCHÖLKOPF *et al.*, 2002)é um algoritmo de aprendizado de máquina supervisionado, geralmente aplicável em problemas de classificação. Dado um conjunto de dados previamente categorizado, o algoritmo recebe um novo dado e atribui á uma categoria. O objetivo principal do *Support vector machine* (SVM) é determinar o melhor hiperplano de separação que represente os dados.

Figura 4 – Hiperplanos de representação



Fonte: (Support Vector Machine, 2022)

Na Figura 4 temos pontos de duas categorias, pontos vazios e preenchidos, esses pontos são representados por retas H_1,H_2 e H_3, olhando para as retas percebe-se que H_1 não representa bem os dados, já as retas H_2 e H_3 separam bem os dados, no entanto, se um novo ponto for adicionado H_2 provavelmente não classificará como a reta H_3, pois, H_3 deixa uma margem maior entre os pontos, então H_3 seria o hiperplano ótimo.

2.6.3 Logistic Regression

Logistic Regression é um algoritmo capaz de lidar com problemas de classificação, podendo ser utilizado para predizer a probabilidade de uma amostra em uma classe determinada. Esse algoritmo pode ser representado pela seguinte equação:

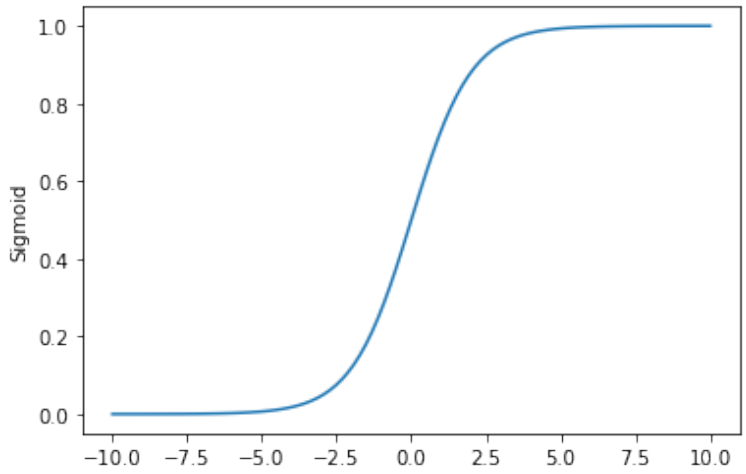
$$H_\Theta(X) = g(X^T\Theta) \tag{2.2}$$

Onde, X representa o vetor de variáveis, sendo esse $(x_0, x_1, \ldots, x_n)$, $\Theta = (\theta_0, \theta_1, \ldots, \theta_n)$ o vetor de parâmetros e g representa a função sigmóide de ativação da *Logistic Regression*. A função de ativação refere-se a curva em forma de S que é capaz de converter qualquer valor que é passado em uma probabilidade que está no intervalo entre 1 e 0. A função sigmóide é referida como uma função de ativação para regressão logística é definida como (KANADE, 2022):

$$g(\bar{x}) = \frac{1}{1+e^{-\bar{x}}} \tag{2.3}$$

Onde, $\overline{x}$ é valor numérico que se deseja transformar e e é a base de logaritmos naturais. A Figura 5 mostra o comportamento da função.

Figura 5 – Função de ativação



Fonte: Autor

Além disso, se a saída da função sigmoide (probabilidade estimada) for maior que um limite predefinido na Figura 5, o algoritmo prevê que a instância pertence a essa classe. Se a probabilidade estimada for menor que o limite predefinido, o modelo prevê que a instância não pertence à classe (KANADE, 2022).

Por exemplo, se a saída da função sigmoide estiver acima de $0,5$, a saída é considerada como 1. Por outro lado, se a saída for menor que $0,5$, a saída é classificada como 0 (KANADE, 2022).

2.6.4 Redes Neurais

Sistemas baseados em inteligência artificial tem demostrado resultados favoráveis e flexíveis, recentemente as Redes Neurais Artificiais (RN) estão ganhando destaque em várias áreas do conhecido, por exemplo, no reconhecimento de padrão, mercado de capitais, distribuição de energia e sistemas especialísticas.As RN são técnicas computacionais que apresentam um modelo matemático baseado na estrutura neural de um ser inteligente, que adquire conhecido por experiência, uma grande rede neural pode ter centenas ou milhares de unidades de processamento(USP, 2009). Elas foram introduzidas por três publicações mais relevantes por (MCCULLOCH; PITTS, 1943),(HEBB, 1949), e (ROSENBLATT, 1958). Apresentando o primeiro modelo de redes neurais, o modelo era básico conhecido como perceptron com aprendizado supervisionado.

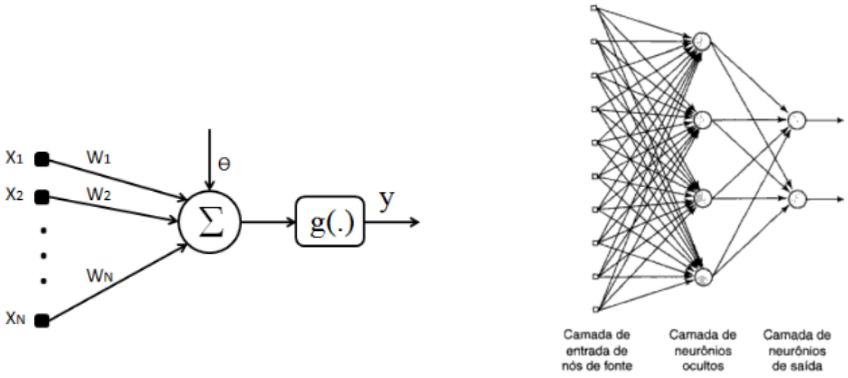
De acordo com (RASCHKA, 2015), o perceptron de camada única com aprendizado supervisionado foi a base para a primeira definição no contexto dessa área, mostrado por Frank Rosenblatt em 1957. Nesta abordagem, teve como objetivo criar um algoritmo inspirado no primeiro modelo de células cerebrais, isto é, o modelo continha ondas nervosas que eram envolvidas no processo de transmissão de sinais químicos e elétrons(MCCULLOCH; PITTS, 1943).

Conforme (BRAGA, 2000), na década de 70, os trabalhos científicos voltados a RN teve um declínio, pois, conforme relatado por (MINSKY; PAPERT, 2017), como só havia um neurônio este não seria capaz de desempenhar tarefas não lineares, levando ao desconhecido, impossibilitando o treinamento do algoritmo com uma ou mais camadas intermediarias.

As RN podem ser estabelecidas como um mapeamento não linear de um vetor de espaço de entrada para um vetor de espaço de saída, que pode ser realizado por meio de camadas de funções de ativação ou neurônios, nos quais coordenadas de entrada são somadas conforme o valor de seus respectivos pesos e “bias” para produzir uma saída simples, ativada ou não, de acordo com o respectivo nível de ativação.

De acordo com (GUYON, 1991), o neurônio é uma estrutura lógica que busca simular a forma, comportamento e funções de um neurônio biológico. Dessa forma, dendritos são trocados por entradas, isto é, ocorre ligações entre célula artificial que são realizadas por meio de elementos chamados de pesos (simulando as sinapses neuronais). Os estímulos são compreendidos pelas entradas e processadas pela função de soma e o bias. O prelúdio de ativação do neurônio biológico foi que deu origem a função de transferência no neurônio artificial (CHUA; YANG, 1988). Esse neurônios compõem uma RN que é mostrado na Figura 6 que é uma representação de uma RN. Em alguns casos, as RN podem possuir diferentes arquiteturas, por exemplo, redes neurais recorrentes, que será retratada a seguir.

Figura 6 – Neurônio artificial e Uma rede neural simples



Fonte: (HAYKIN, 2001)

2.6.5 Redes Neurais Recorrentes

As redes neurais tradicionais não podem lidar com entradas dependentes muito longas, isso requer outro tipo de arquitetura de rede neural (BASHEER; HAJMEER, 2000). Uma rede neural recorrente, essa arquitetura de rede dependendo das interações anteriores na rede, ou seja, possui uma memória para guardar as dependências, isso difere das outras, pois, quando os dados de entradas são sequências, ela se saem melhor (BASHEER; HAJMEER, 2000). Portanto, as aplicações que envolvem dados de entrada como vídeo, som ou texto são adequadas.

A entrada para esta rede é um tipo de sequência, denominado X_t, um exemplo de sequência é uma frase, onde X_t é a representação da t-ésima palavra. A saída da rede é representada por O_t, e os estados ocultos são chamados de h_t. As letras W e b representam o valor do peso e o valor do bias do neurônio artificial, respectivamente, que são obtidos através do treinamento da rede.

Aqui temos o cálculo dos estados ocultos da variável oculta do passo de tempo atual é determinado pela entrada do passo de tempo atual junto com a variável oculta do passo de tempo anterior.

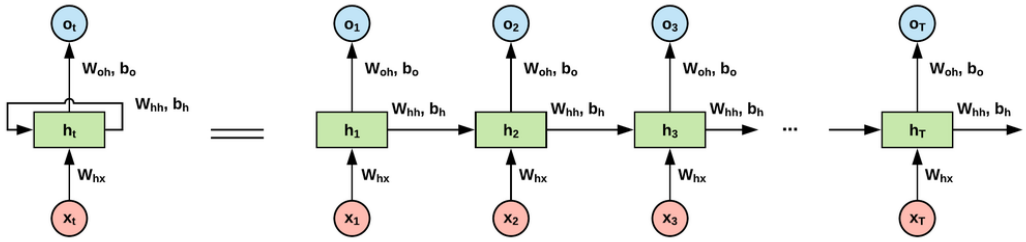
$$h_t = \sigma(X_t W_{xt} + h_{t-1} W_{hh} + b_h) \tag{2.4}$$

Onde, σ é a função de ativação. A camada de saída é dada por:

$$O_t = h_t W_{hq} + b_q \tag{2.5}$$

Na Figura 7 mostra o que foi descrito anterior.

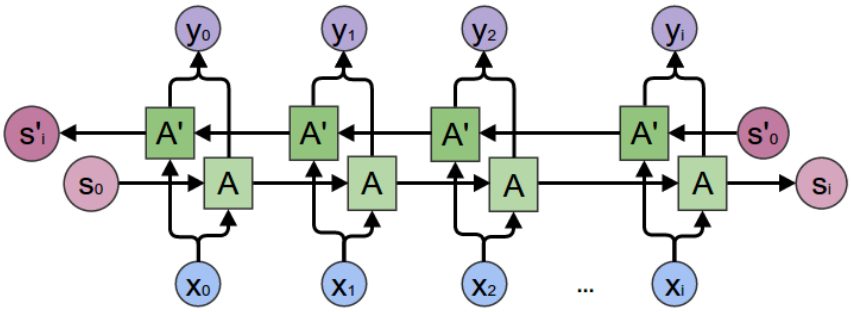
Figura 7 – Arquitetura de Rede Neural Recorrente Simples



Fonte: (GUDIKANDULA, 2019)

Redes recorrentes são usadas em textos, pois, conseguem ter uma performance muito boa. Um exemplo clássico é classificação de texto para descobrir um certo interesse de um grupo ou classificar artigos de notícias em categorias. Também é interessante cita as redes recorrentes bidirecionais que será apresentada na Figura 8. A entrada dessa rede é representada por X, onde ele representa as palavras de uma sentença, X alimenta duas redes ao mesmo tempo, uma A e A', uma avalia a sentença em uma ordem de leitura, enquanto a outra avalia em uma ordem inversa de leitura(GUDIKANDULA, 2019). A saída para essa rede é a concatenação de ambas direções dos resultados obtidos. Assim obtendo a relação entre as palavras da sentença que precedem e as que sucedem, dando um contexto legível do sentido da sentença (GUDIKANDULA, 2019).

Figura 8 – Arquitetura de Rede Neural Recorrente Bidirecional



Fonte: (LEE, 2017)

2.6.6 Transformers

Transformers é um algoritmo de Aprendizado Profundo adotado em 2017, com finalidades no processamento de linguagem natural. Ele é um algoritmo que usa a atenção para aumentar a velocidade para agilizar o treinamento. *Transformers* superam os algoritmos de tradução automática neural do Google em tarefas específicas. O maior benefício, entretanto, vem de como o *transformers* se presta à paralelização. O *transformers* é formado por *encoders*, *decoders* que contém *self-attention* e *Feed forward* que é empilhada em camadas totalmente

conectadas. Como mostra na Figura 9:

Figura 9 – Transformers.

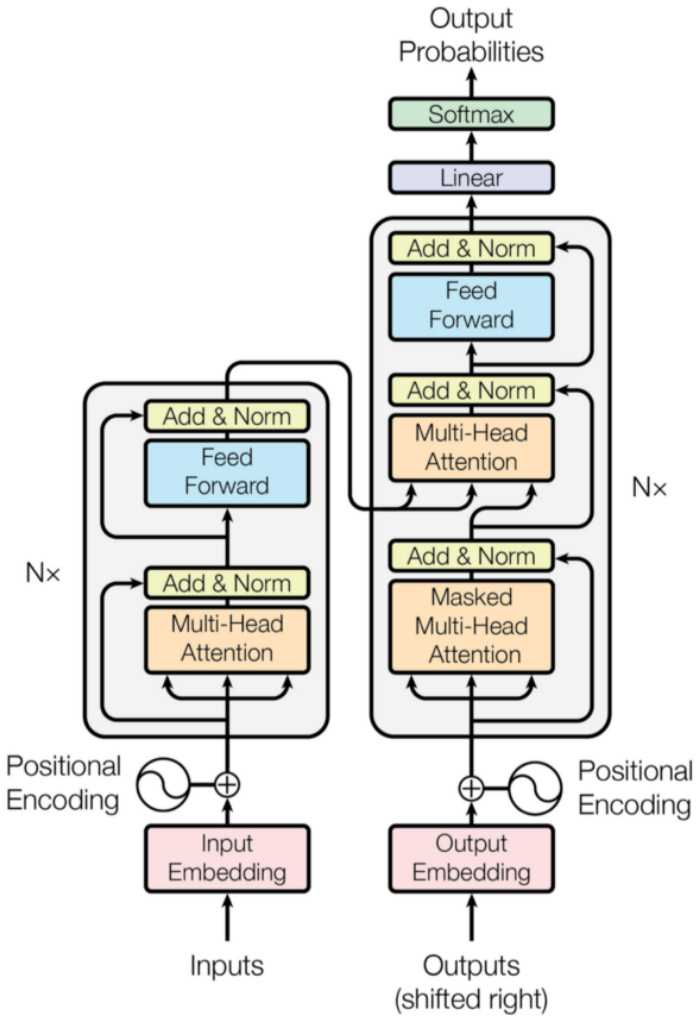


Figure 1: The Transformer - model architecture.

Fonte: (VASWANI *et al.*, 2017)

O algoritmo é composto por dois módulos, que são responsáveis por diversas tarefas. Estes módulos *encoder* e *decoder*, cada um desses módulos com 6 pilhas, estas são compostas por: *self-attention* e *feed forward*, cada *self-attention* é um mecanismo de atenção que relaciona diferentes posições de uma sequência, com objetivo de calcular uma representação. *Feed forward* é responsavel por converter as informações vindas da camada *self-attention* em *embedding* de tamanho menor. De forma análoga o decodificador é semelhante ao codificador, também com 6 pilhas, uma diferença é que entre essas camadas têm uma camada adicional conhecida como *encoder-decoder attention*(VASWANI *et al.*, 2017).

O codificador é composto por uma pilha de 6 camadas idênticas. Cada camada

> possui duas subcamadas. O primeiro é um mecanismo de *felf-attention* com *multi-head* e o segundo é uma rede *feed-forward* simples e totalmente conectada no sentido da posição. Empregamos uma conexão residual em torno de cada uma das duas subcamadas, seguida pela normalização da camada. Ou seja, a saída de cada subcamada é $LayerNorm(x + Sublayer(x))$, onde $Sublayer(x)$ é a função implementada pela própria subcamada. Para facilitar essas conexões residuais, todas as subcamadas do algoritmo, tal como as camadas de incorporação, produzem saídas de dimensão $d_{model} = 512$. (VASWANI *et al.*, 2017)

> O decodificador também é composto por uma pilha de 6 camadas idênticas. Além das duas subcamadas em cada camada do codificador, o decodificador insere uma terceira subcamada, que executa a *multi-head attention* sobre a saída da pilha do codificador. Semelhante ao codificador, empregamos conexões residuais em torno de cada uma das subcamadas, seguidas pela normalização da camada. Também modificamos a subcamada de *self-attention* na pilha do decodificador para evitar que as posições atendam às posições subsequentes. Esse mascaramento, combinado com o fato de que os *embeddings* de saída são deslocados em uma posição, garante que as previsões para a posição *i* possam depender apenas das saídas conhecidas em posições menores que *i*. (VASWANI *et al.*, 2017)

O algoritmo para fazer a codificação deve aplicar um método chamado *positional encoding* das diferentes palavras. Como não há uma rede recorrente para lembrar de como as sequências são alimentadas no modelo, é necessário dar a cada palavra uma posição relativa na sequência da frase, por se tratar de uma sequência a ordem é importante. Essas posições são inseridas em um vetor com d vetor dimensional para representar o vetor de palavras. Dito de oura forma, aprimora-se a entrada do algoritmo para injetar a ordem das palavras. Formalmente, pode ser definido como sendo t uma posição desejada em uma frase da entrada, $\vec{pt}$ seja sua posição corrente e d a dimensão da codificação, então $f : \mathbb{N} \to \mathbb{R}^d$ irá nos fornecer uma saída $\vec{pt}$.

$$\vec{pt}^{(i)} = f(t)^{(i)} = \begin{cases} \sin(w_k \cdot t), & \text{se } i \text{ é par} \\ \cos(w_k \cdot t), & \text{se } i \text{ é ímpar} \end{cases} \tag{2.6}$$

onde

$$w_k = \frac{1}{10000^{\frac{2k}{d}}} \tag{2.7}$$

Como podemos observar as frequências estão decaindo ao longo da dimensão do vetor, portanto, a função forma uma progressão geométrica de 2π para $10000 \cdot 2\pi$. Com essa representação posicional é possível utilizar uma função de atenção para descrever como o mapeamento de uma consulta e de um conjunto de pares de valores-chave para uma saída, onde a consulta, as chaves, os valores e a saída são todos vetores. A saída é calculada como uma

soma ponderada dos valores, onde o peso atribuído a cada valor é calculado por uma função de compatibilidade da consulta com a chave correspondente(VASWANI *et al.*, 2017).

Figura 10 – Os dois tipos de atenção.

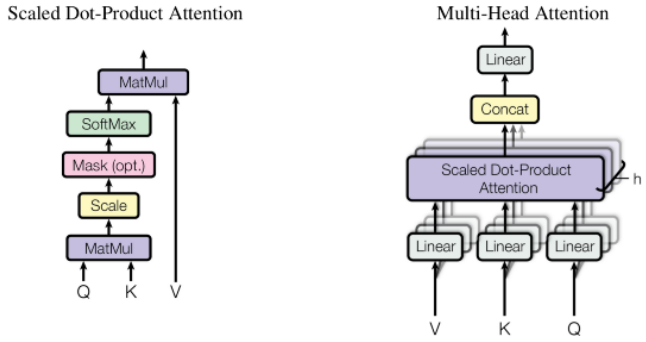


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

Fonte: (VASWANI *et al.*, 2017)

No lado esquerdo da Figura 10 tem "Scaled Dot-Product Attention". Onde há uma entrada que consiste em consultas e chaves da dimensão d_k, valores de dimensão d_v, aonde é calculado o produto escalar da consulta com todas as chaves, sendo divido cada uma por $\sqrt{d_k}$, aplicando a função de *softmax* para extrair os pesos dos valores. De forma prática a função é calculada em conjunto de consultas simultaneamente, agrupada em uma matriz Q. Da mesma forma para as chaves e valores que são compactadas em matrizes K e V(VASWANI *et al.*, 2017). A matriz resultado da atenção pode ser resumida na equação abaixo:

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.8}$$

No lado direito da Figura 10 tem o mecanismo de *Multi-Head Attention* que em vez de usar apenas uma função de atenção com chaves dimensionais de algoritmo, valores e consultas, também é benéfico projetar linearmente as consultas das chaves e valores em h vezes com diferentes projeções lineares aprendidas para dimensões d_k, d_k e d_v, respectivamente. Em cada uma dessas versões projetadas nas consultas, chave e valores são realizadas em uma função de atenção em paralelo, gerando valores de saída d_v dimensões. Estes são concatenados e mais uma vez projetados, resultando nos valores finais. *Multi-Head Attention* permite que o algoritmo atenda conjuntamente às informações de diferentes subespaços de representação em diferentes

posições. Com uma única cabeça de atenção, a média inibe este(VASWANI *et al.*, 2017).

$$\begin{aligned} MultiHead(Q,K,V) &= Concat(head_1,\ldots,head_h)W^0 \\ onde \quad ,head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.9}$$

Transformers é um algoritmo que é geralmente usado no campo de processamento de linguagem natural, por exemplo, em tarefas de tradução automática, previsão de próxima sentença, análise de sentimento e previsão de séries temporais. Além disso, (MAXIME, 2019) diversos algoritmos usam *transformers* como recursos para melhorar seus desempenhos em tarefas de estado da arte. Um exemplo, é o (DEVLIN *et al.*, 2018) BERT um algoritmo de representação de linguagem do Google AI, que usa pré-treinamento e ajuste fino para criar algoritmos de ponta para uma ampla gama de tarefas. Essas tarefas incluem sistemas de resposta a perguntas, análise de sentimentos e inferência de idioma, usando *transformers* para fazer essas diversas tarefas.

2.6.7 BERT

É um algoritmo usado para o pré-treinamento em processamento de linguagem natural criado pelos pesquisadores do *google AI language*. O BERT foi publicado em 2018 por jacob Devlin e seus colegas. O algoritmo que mostrou resultados promissores em 11 tarefas de linguagem natural.

O algoritmo tem suas raízes em representações contextuais com pré-treinamento, que incluem o aprendizado semi-supervisionado de sequências, como pré-treinamento generativo, com ELMo (PETERS *et al.*, 2018) e ULMFit (HOWARD; RUDER, 2018). Além disso, o BERT difere dos algoritmos anteriores por ser uma representação de linguagem profunda bidirecional e não supervisionada, o algoritmo leva em consideração o contexto das palavras da sentença. Por exemplo, enquanto o vetor para "correr" terá a mesma representação vetorial word2vec para ambas as ocorrências nas frases "Ele está administrando uma empresa" e "Ele está administrando uma maratona", o BERT vai olhar a frase como um todo é fazer uma incorporação correta para cada palavra relacionada.

O BERT utiliza *transformers* para entender a semântica de uma frase, *transformers* é um mecanismo que aprende relações contextuais entre palavras em um documento. Em sua forma simples, o *transformers* inclui dois mecanismos separados — um codificador que lê a entrada de texto e um decodificador que produz uma previsão para a tarefa (HOREV, 2018).

Para o BERT apenas o mecanismo de codificação é necessário, como o ele tem como objetivo gera um algoritmo de linguagem, ao contrário dos algoritmos direcionais que leem a entrada de texto de forma sequencial, já o codificador *transformers* faz a leitura na sequência inteira das palavras de uma vez só. Sendo considerado bidirecional, embora seja mais exato dizer que não é direcional. Essa característica permite que o algoritmo aprenda o contexto de uma palavra com base em todos os seus arredores(HOREV, 2018).

O algoritmo para fazer essa compreensão faz uso de duas estrategias de treinamento *Masked LM* (MLM) e *Next Sentence Prediction* (NSP) :

MLM: Para treinar uma representação bidirecional profunda, simplesmente é trocado alguma porcentagem dos *tokens* de entrada aleatoriamente e depois prever esses *tokens* mascarados. Neste caso, os vetores ocultos finais correspondentes aos *tokens* da máscara são alimentados em um *softmax* de saída sobre o vocabulário, como em um MLM padrão. Em todos os experimentos, mascaramos 15% de todos os itens do *WordPiece* em cada sequência aleatoriamente. Em contraste com os codificadores automáticos (DEVLIN *et al.*, 2018). O algoritmo tenta prever o valor original das palavras mascaradas, com base no contexto fornecido pelas outras palavras não mascaradas na sequência. As palavras em cada sequência que foram mascaradas são substituídas por um *token MASK* ,de forma aleatorizada, para que não haja problema no ajuste fino(DEVLIN *et al.*, 2018).

NSP: O algoritmo na fase de treinamento recebe pares de frases como entrada, com base nelas prevê se a segunda frase e a continuação da primeira. No pré-treinamento 50% das segundas frases são continuações da primeira, 50% delas são rotuladas como não texto, ou seja, não é uma continuação da primeira frase. Para ajudar o algoritmo a distinguir entre às duas frases no treinamento, a entrada é processada da seguinte maneira antes de entrar no algoritmo: Um *token* inserido no começo da sentença ([CLS]) e um *token* inserido no final da sentença ([SEP]) é inserido no final de cada frase. Com essas duas estratégias o BERT é capaz de relacionas as palavras da frase com uma ótima semântica de entendimento, o que facilita no processo de compreensão de grandes documentos ou pequenos (DEVLIN *et al.*, 2018).

O algoritmo pode ser utilizado para uma gama de tarefas diferentes de linguagem, enquanto adiciona apenas uma pequena camada ao algoritmo principal: as tarefas de classificação, como a análise de sentimentos, são executadas de maneira semelhante à classificação da próxima sentença, adicionando uma camada de classificação na parte superior da saída do *transformers* para o *token* [CLS], nas tarefas de Resposta a perguntas (por exemplo, SQuAD v1.1), o *software*

recebe uma pergunta sobre uma sequência de texto e é necessário para marcar a resposta na sequência. Usando o BERT, um modelo de perguntas e respostas pode ser treinado aprendendo dois vetores extras que marcam o início e o fim da resposta (HOREV, 2018). Após treinamento, o ajuste fino é direto. Para isso são necessárias 4 etapas:

1. pares de frases em parafraseando
2. pares hipótese-premissa em vinculação,
3. pares de passagem de pergunta em resposta de pergunta
4. um par de texto degenerado na classificação de texto ou marcação de sequência

Comparado ao pré-treinamento, o ajuste fino é relativamente barato, pois pode ser feito em algumas horas com GPU (DEVLIN *et al.*, 2018).

3 TRABALHOS RELACIONADOS

Neste capítulo, serão abordados alguns trabalhos que usaram algoritmos de Aprendizado de Máquina e Aprendizado Profundo com ênfase na detecção automática de *fake news*, como (CRUZ *et al.*, 2019), (MIHALCEA; STRAPPARAVA, 2009), (YANG *et al.*, 2018), (PÉREZ-ROSAS *et al.*, 2017), (MARKINES *et al.*, 2009)

(CRUZ *et al.*, 2019) teve por objetivo focar o uso de técnicas de transferência de aprendizagem nos modelos de linguagem para detecção automática de *fake news*. Os modelos são: BERT, ULMFIT e GPT-2. Para demostra a contribuição com TC, foram usados modelos-base com técnicas de aprendizagem máquina para compará-los com os modelos com TC. O conjunto de dados usado no experimento foi extraído de fontes do mundo real e da *web*. Os artigos de notícias falsas foram adquiridos em sites marcados como notícias falsas pela organização independente sem fins lucrativos de verificação de fatos, a Verafiles, e pela União Nacional de Jornalistas nas Filipinas (NUJP). Os artigos de notícias reais foram adquiridos nos principais sites de notícia das Filipinas, incluindo Pilipino Star Ngayon e Abante.

O aprendizado multitarefa, mostra que o aumento das técnicas de transferência baseadas em transformadores com perdas de modelagem de linguagens auxiliares melhora seu desempenho adaptando-se à estilometria. Com isso, melhorando o desempenho da TC em $4-6\%$, atingindo uma precisão de 96% no melhor modelo.

(MIHALCEA; STRAPPARAVA, 2009) teve por objetivo o uso de técnicas padrões de processamento de linguagem natural para a detectar linguagem enganosa. Dois algoritmos de aprendizagem de máquina são usados para os experimentos: *Naives bayes* e SVM. O objetivo é a detecção automática de linguagem enganosa em três tópicos diferentes como aborto, pena de morte, melhor amigo. Os dados que contém cada um dos tópicos foram especialmente para pesquisa (MIHALCEA; STRAPPARAVA, 2009). Para isso, contaram com a participação de voluntários para que formassem uma opinião sobre cada tópico: aborto, pena de morte, melhor amigo. Para cada um dos três tópicos, usaram um serviço de anotação de tarefa(*Amazon Mechanical Turk*). Foi pedido que eles elaborassem dois tipos de opiniões uma verdadeira e, outra falsa. Para isso coletaram 100 amostras verdadeiras e falsas de cada tópico para a análise.

Para esse experimento, foi utilizado as classes de palavras definidas na Pesquisa Linguística e Contagem de Palavras (LIWC), que foi desenvolvida como um recurso para a análise linguística. A versão 2001 do LIWC inclui cerca de 2.200 palavras e hastes de palavras

agrupadas em cerca de 70 categorias principais relevantes para os processos psicológicos (por exemplo, EMOÇÃO, COGNIÇÃO). O LIWC léxico foi validado, mostrando uma correlação significativa entre classificações humanas de um grande número de textos escritos e a classificação obtida através de análises dos mesmos textos com base na LIWC. O resultado desse experimento mostra que o modelo que se saiu melhor com esta abordagem foi o Naıve Bayes com uma precisão de 59.8% no treino-teste. Os métodos adotados por este trabalho fornecem uma visão sobre como devemos analisar textos com um validado de léxico.

(YANG *et al.*, 2018), propôs um modelo unificado para detectar notícias falsas automático em textos e imagens, usando as redes neurais convolucionais para a classificação. O conjunto de dados desse experimento foi extraído de textos e metadados de mais de 240 sites da *Megan Risdal* no *kaggle*, para notícias falsas. Quanto às notícias reais são rastreadas a partir dos sites oficias de notícias reais, como o *New York Times*, *The Washington Post*, e entre outros. Para a análise foi considerado apenas título, texto, e as informações da imagem.

O Modelo unificado que combina as informações de texto e imagem com recursos explícitos e latentes, tem forte capacidade de expansão, o que mostra que características diferentes podem ser facilmente adaptada ao modelo. Além disso, a rede neural convolucional cria o modelo para ver a entrada de uma só vez, e pode ser treinada mais rápido que o *Long Short Term Memory* (LSTM) e diversos outros modelos de RN. O resultado do experimento mostra que o modelo unificado pode detectar notícias falsas com sucesso, tendo um nível de acurácia de $92,2\%$. O conjunto de dados analisado tem enfoque nas eleições presidências americanas. Além disso, demostra a relevância em analisar títulos de manchetes, tópicos, textos, imagens na identificação de notícias falsas.

(PÉREZ-ROSAS *et al.*, 2017), propôs recursos e modelos computacionais para a tarefa de detecção de *fake news*. Para isso foram usados as seguintes técnicas linguísticas, como: *Ngrams, Punctuation, Psycholinguistic features, Readability, Syntax*. Os dois conjuntos de dados apresentados cobrem vários domínios diferentes. O primeiro conjunto de dados foi coletado do *crowdsourcing* cobrindo seis domínios de notícias (por exemplo, negocios, educação). O segundo conjunto de dados é obtido diretamente da web e abrangendo notícias de celebridades. Primiero, foi coletado um conjunto de dados de notícias legítimas pertencentes a, seis domínios diferentes (esportes, negócios, entretenimento, política, tecnologia e educação). As notícias verdadeiras são provenientes de sites de notícias predominantes nos EUA, como ABCNews,

CNN, etc. Já para a versão falsa de notícias foi usado *Amazon Machine Turk*. No segundo conjunto de dados, voltado a celebridades, foi coletado de revistas online como *Entertainment Weekly*, *People Magazine* entre outras.

No experimento, foi realizado várias experiências com diferentes combinações de conjuntos de recursos para explorar suas previsões de forma independente e conjunta. Utilizando um classificador SVM linear, fazendo uma validação cruzada 5, vezes, para medir o desempenho de cada modelo. Para isso vários algoritmos de *marchine learning* são usados para a detecção de *fake news*. A maioria dos classificadores obtém desempenhos bem acima da linha de base, o que indica que a tarefa de detecção de notícias falsas pode ser efetivamente abordada usando linguagem. O resultado do experimento sugere que a abordagem computacional linguística auxilia no procedimento de identificação de notícias falsas de maneira automatizada. A abordagem proposta pelo artigo, sugere que, para diferenciar entre conteúdo falso e genuíno, vale a pena examinar o nível lexical, sintático e semântico de um item de notícia em questão. O desempenho do sistema é similar ao dos humanos na área de trabalho, com uma precisão de até 76% de acurácia.

(MARKINES *et al.*, 2009), teve por objetivo a identificação automática de notícias falsas, usando modelos de linguagem pré-treinados: *Multi-head* LSTM, *Bidirectional* LSTM *concatenated*. O conjunto de dados é provido do FNC-1, esse conjunto de dados já está rotulado como notícias falsas ou notícias verdadeiras. O *dataset* para esse artigo é divido em 60% treino, 20% validação, 20% teste. O modelo *Bidirectional* LSTM *concatenated*, usa dados pré-treinados do Google gerando dois vetores de incorporação que são concatenados para alimentar o modelo, esse modelo usa duas *convolutional neural network* (CNN)s. O outro modelo também usa vetores pré-treinadas do google usando duas camadas da CNN.

O resultado desse experimento mostra que a identificação automática de *fake news* usando técnicas de aprendizagem de máquina, principalmente aprendizagem profunda e redes neurais. Mostra que modelos podem detectar muito bem notícias falsas, os modelos que se destacaram nessa pesquisa foram LSTM bidirecional concatenado, e LSTM de várias cabeças. Esses modelos foram aplicados ao conjunto de dados do FNC-1 e os resultados mostraram que o Bidirecional o modelo concatenado LSTM tem a mais alta precisão com 85% seguido pelo modelo LSTM *Multi-head*de cerca de 83%. Em precisão o modelo LSTM é mais preciso que LSTM *Multi-head*. Segundo essa pesquisa seria mais interessante usar o modelo LSTM

Multi-head.

(ZHANG *et al.*, 2018) teve por objetivo criar um modelo de rede difusa profunda chamada de *FAKEDETECTOR*, baseando-se em um conjunto de recursos explícitos e latentes retirados de informações textuais. O conjunto de dados usado neste artigo inclui os *tweets* publicados pelo *PolitiFact* em sua conta oficial no Twitter, bem como os artigos de verificação de fatos escritos sobre essas declarações no site do *PolitiFact*. O *PolitiFact* também categoriza as declarações em grupos diferentes sobre os assuntos, o que denota os tópicos sobre os quais essas declarações se referem. Com base na credibilidade das declarações feitas pelos criadores, o *PolitiFact* também fornece a avaliação de credibilidade para esses criadores e sujeitos. O *FAKEDETECTOR* constrói um modelo de rede difuso profundo para aprender as representações de artigos de notícias, criadores e assuntos simultaneamente. O *FAKEDETECTOR* apresenta dois componentes principais: aprendizado de recursos de representação e inferência de etiquetas de credibilidade, que juntos comporão o modelo de rede difusa. Com base nas conexões entre artigos de notícias, criadores e assuntos de notícias; foi proposto um modelo de rede difuso profundo para incorporar as informações da estrutura de rede no modelo de aprendizado. Nesse artigo, também foi introduzido um novo modelo de unidade difusiva, o GDU. O Modelo GDU aceita várias entradas de diferentes fontes simultaneamente e pode efetivamente fundida para a geração de saída com conteúdo "esquecer" e "ajustar" as portas. Experimentos extensivos realizados em um conjunto de dados de notícias falsas do mundo real, ou seja, o *PolitiFact*, demonstraram o desempenho excepcional do modelo proposto na identificação de artigos de notícias falsas, criadores e sujeitos da rede.

(BUNTAIN; GOLBECK, 2017) propôs automatizar a detecção de notícias falsas nos dados do *twitter*, realizando um estudo da precisão em dois conjuntos de dados do *Twitter* focados na credibilidade: – *CREDBANCK* e *PHEME*. Aplicaram esses métodos ao conteúdo do Twitter proveniente do conjunto de dados de notícias falsas do *BuzzFeed* e mostraram que os modelos treinados contra trabalhadores terceirizados superam os modelos com base na avaliação de jornalistas e modelos treinados em um conjunto de dados. Todos os três conjuntos de dados, alinhados em um formato uniforme, também estão disponíveis ao público.

Cada um dos três conjuntos de dados é separado uniformemente, permitindo assim que investigação de destaque seja realizada. Uma analise característica diz que alguns

recursos são mais preditivos para avaliação do jornalismo, resultados nos quais são providos dos trabalhos anteriores. Para prever a precisão que se enquadram em quatro tipos: estrutural, usuário, conteúdo e temporal. Dessas características, incluíram catorze das características mais importantes encontradas. Omitindo as duas características nos *links* da *Web* mais frequentes. Os recursos estruturais capturam propriedades específicas do Twitter do fluxo de *tweets*, incluindo distribuição de volume e atividade de *tweet* (por exemplo, proporções de compartilhamentos de mídia de *retweets*). Os recursos do usuário capturam propriedades dos autores, como interações, idades da conta, contas de amigo / seguidor e status verificado no Twitter. Medida de recursos de conteúdo, aspectos textuais dos tweets, como polaridade, subjetividade e concordância.Este trabalho demonstra um sistema automatizado para detectar notícias falsas em *threads* populares do *Twitter*. Esse sistema pode ter um valor inestimável para os usuários de mídia social, aumentando e apoiando seus próprios julgamentos de credibilidade, o que seria um benefício crucial, considerando as fraquezas.

(KALIYAR, 2018), teve como foco mostrar diversas abordagens na detecção de *fake news*. Utilizando técnicas de processamento de linguagem natural, aprendizado de máquina e aprendizado profundo para classificar os conjuntos de dados. Realizaram uma auditoria dos trabalhos de detecção de *fake news*, explicando os modelos e o comportamento de cada um. Também exploraram os benefícios da extração de recursos, recursos como n-grama, recursos *term frequency-inverse document frequency* (TF-IDF) foram extraídos e usados em modelos. Também exploramos a eficácia dos recursos de incorporação de palavras. Essa pesquisa fornece uma visão dos melhores modelos, o que é bastante útil.

(AMINE *et al.*, 2019), propôs um modelo de aprendizagem profunda que detecta notícias falsas em diferentes tipos de característica. O conjunto de dados proposto por *kaggle*, contém cerca de 20.000 artigos rotulados para duas classes: rótulo 10000 falso e real. Os seguintes atributos estão incluídos: id: *unique id for a news article, title: the title of a news article, author: author of the news article, text: the text of the article; could be incomplete, label: a label that marks the article as potentially fake (1)or real (0)*. Para isso, usaram expressões regulares para remover números irrelevantes do texto e interromper as palavras para evitar ruídos de recursos no processo de classificação do texto. Usaram técnicas de incorporação de palavras em redes neurais convencionais para extrair recursos baseados em texto e compararam diferentes

arquiteturas de aprendizado profundo enquanto mesclavam duas CNNs com diferentes metadados (texto, título e autor), obtendo um modelo com uma precisão de até 97% com apenas texto e autor.

(LIANG *et al.*, 2018), teve por objetivo a detecção de *clickbait* em outro idioma com base no modelo treinado em um idioma já. *Clickbait* são manchetes de notícias que exageram os fatos ou ocultam manchetes de fatos parciais. As estatísticas mostram que os *clickbaits* são predominantes em todos os idiomas. No entanto, pesquisas anteriores sobre detecção de *clickbait* se concentram principalmente no inglês. Inspirado por isso, propuseram que o aprendizado de transferência seja aplicado para transferir o modelo na detecção de *clickbait* de um idioma de origem para outros idiomas. Para isso treinaram um modelo de origem no corpus em inglês e o transferiram para o corpus em chinês.

O modelo com poucas anotações, foi capaz de obter um desempenho comparável ao modelo que utilizou muitas anotações, mostrando o poder da transferência de aprendizado. Os resultados dos experimentos mostram que o modelo de aprendizado de transferência neste documento pode obter desempenho semelhante no idioma de destino usando menos anotações, mostrando a eficácia e a robustez desse modelo. O modelo obteve um nível de acurácia bem alto de 94%. O que mostra que podem reaproveitar modelos de detecção de *fake news* em outros idiomas.

Com os métodos descritos, fica claro que alguns perdem ao tentar generalizar apenas para tipos específicos de *fake news* ou *dataset* específicos. No entanto, como no estudo (MIHALCEA; STRAPPARAVA, 2009) classificando notícias falsas específicas de domínio, outras abordagens mostraram que os modelos generalizam bem, dependendo das técnicas usadas neles. Assim como a abordagem (CRUZ *et al.*, 2019) que pode ser usada como base para outras línguas, a abordagem usada por (CRUZ *et al.*, 2019) para realizar experimentos sustenta a ideia de usar uma abordagem semelhante no corpus criado por (MONTEIRO *et al.*, 2018) e fornecer melhores resultados em aplicações de transferência de conhecimento. Por fim, outros métodos fornecem informações sobre recursos relevantes na análise de texto. Todos os procedimentos descritos são fundamentais para que os resultados finais sejam satisfatórios. Também é importante ressaltar que todos os procedimentos abordados usam *dataset* diferentes como descrito na Tabela 2. Entretanto, as formas de validar os resultados são praticamente as mesmas descritas na Tabela 3

Tabela 2 – Trabalhos relacionados e suas base de dados

Tabela dos trabalhos relacionados	
Datasets	Artigos Relacionados
Fake news Filipino	Localization of Fake News Detection via Multitask Transfer Learning
Reconhecime de linguagem enganosa	The Lie Detector: Explorations in the Automatic Recognition of Deceptive Language
Getting Real about Fake News	TI-CNN: Convolutional Neural Networks for Fake News
FakeNewsAMT/Celebrity	Automatic Detection of Fake News
FakeNewsChallenge(fnc-1)	Automatic Identification of Fake News Using Deep Learning
real-world fake news	Fakedetector: Effective fake news detection with deep diffusive neural network
Credbank-data/PHEME	Automatically identifying fake news in popular twitter threads
Fake News	Merging deep learning model for fake news detection
click-bait in English(Chakraborty)	Sentence Classification with Transfer Network
Fake br	Contributions to the Study of Fake News in Portuguese: New Corpus and Automatic Detection Results

Fonte: Autor

Tabela 3 – Métricas dos trabalhos relacionados

Tabela de métricas				
Trabalhos Relacionados	F1-score	Precision	Recall	Acuracy
Localization of Fake News Detection via Multitask Transfer Learning				91,59
The Lie Detector: Explorations in the Automatic Recognition of Deceptive Language				57,8
TI-CNN: Convolutional Neural Networks for Fake News	92,1	92,2	92,77	
Automatic Detection of Fake News				76
Automatic Identification of Fake News Using Deep Learning	76,36	77,11	75,61	85,3
Fakedetector: Effective Fake news detection with deep diffusive neural network	84,3	68,3	98	71,6
Automatically identifying fake news in popular twitter threads				73,80
Merging deep learning model for fake news-detection	97	98	94	96
Sentence Classification with Transfer Network	93	93	93	94
Contributions to the Study of Fake News in Portuguese: New Corpus and Automatic Detection Results	89	89	89	89

Fonte: Autor

4 METODOLOGIA

Neste capítulo será detalhado os métodos e as técnicas que serão utilizadas para o desenvolvimento deste trabalho. Na seção 4.1 , serão descritos os procedimentos realizados durante o desenvolvimento do projeto. Cada etapa será abordada, no intuito de esclarecer os métodos, ferramentas a recursos a serem utilizados. Na seção 4.2 as etapas serão descritas para fornece o entediamento das técnicas a serem usadas para a criação da hipótese.

4.1 Visão geral

O objetivo deste trabalho é identificar a categoria que distingue notícias verdadeiras de notícias falsas usando de técnicas de aprendizado profundo e aprendizado de máquina. Partindo da premissa que o conteúdo das *fake news* são diferentes do conteúdo jornalístico e, tentam manipular emocionalmente o leitor para ter viés em algum assunto. Espera-se que os modelos sejam capazes de diferenciar conteúdo falso do verdadeiro a partir do reconhecimento das características que esses textos têm, usando apenas a síntese da notícia com mínimo de pré-processamento dos dados.

Um dos primeiros passos foi a coleta dos dados do corpus *Fake.br* para treinar e validar os modelos propostos. Para isso, foi necessário realizar uma exploração dos dados, para entender o comportamento deles e obter os dados relevantes para a extração de características.

A próxima etapa foi fazer a escolha de modelos que melhor se ajusta para a tarefa de classificação automática. Para esta etapa foi escolhido alguns modelos de aprendizado clássico, aprendizado profundo e aprendizado profundo com técnicas de transferência de conhecimento. Os modelos são Naive Bayes, regressão logística, support vector machine, rede neural recorrente do tipo LSTM e BERT. Cada modelo exige que os dados tenham um tipo de representação especifica na entrada. Para os modelos de aprendizagem clássicos os dados de entrada serão do tipo uma representação estatística e para os modelos de aprendizado profundo a representação será sequencial.

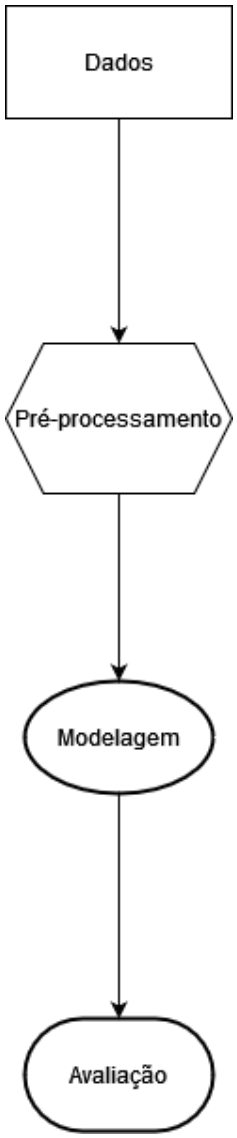
Para avaliar os modelos, foram utilizados as métricas de avaliação usadas na literatura; as mesmas são: *accuracy*, *precision*, *recall* e *f-mensure*.

Ao final deseja-se identificar as melhores abordagens baseadas em modelos que melhor classificam as características. Para isso, os modelos devem ter seus atributos ajustados para que possam se adaptar aos dados de forma positiva ou negativa.

4.1.1 Fluxograma da proposta

Na Figura abaixo ilustra o fluxo do desenvolvimento dos modelos.

Figura 11 – fluxo dos passos da proposta



Fonte: Autor

4.2 Métodos e Modelos

Nesta seção, serão descritos os procedimentos realizados durante o desenvolvimento do projeto. Cada etapa será abordada, no intuito de esclarecer os métodos, ferramentas e recursos a serem utilizados. O autor partiu da hipótese de que é possível classificar notícias falsas automaticamente usando métodos de aprendizado profundo com transferência de conhecimento, devido a diferenças na forma de escrita entre essas notícias e conteúdo verdadeiro e este trabalho

será realizado com intuito de testar a hipótese proposta. Para isso, o projeto foi dividido nas seguintes atividades:

1. Revisão bibliográfica sobre temas *fake news*, aprendizado profundo, transferência de conhecimento e uso de aprendizado de máquina para detecção de *fake news*;
2. Descrição da base de dados que será usada para treinamento e validação dos modelos;
3. Investigar técnicas de transferência em modelos de aprendizado profundo e em modelos que não usam elas;
4. Explorar métodos de aprendizado tradicional que fazem a classificação automática de *fake news* em textos;
5. Implementar os modelos propostos para demostrar a diferença entre eles;
6. Realizar o treinamento e validação dos modelos propostos;
7. Fazer análise dos resultados obtidos pelos modelos depois da validação;

4.2.1 Conjunto de Dados

Depois de muita procura por uma base de dados em português brasileiro, enontrou-se o corpus Fake.br produzido pelo projeto Opinando do Núcleo Interinstitucional Linguística Computacional ou NILC compila 7.200 notícias em língua portuguesa, das quais 3.600 são falsas 3600 reais, diversos tipos (MONTEIRO *et al.*, 2018). Corpus ainda está disponível Use metainformações sobre cada elemento, como autor, link para obtê-lo, gênero e outros Métricas semânticas e sintáticas (HARTMANN *et al.*, 2017). O NILC consiste em pesquisas na área de computação no Institute of Mathematics and Computer Science (ICMC) USP São Carlos e linguistas da Universidade Federal de São Carlos (UFSCar), que Universidade Estadual Paulista (UNESP) e Universidade Estadual de Maringá (UEM). O conjunto de dados que será utilizado neste trabalho contem várias notícias que foram extraídas de várias fontes, como A Folha do Brasil, The Jornal Brasil, G1 e Folha de São Paulo. Neste trabalho será utilizado o *Fake.Br Corpus* sendo que as notícias foram postas por tamanho e assunto, para não ter uma análise enviesada dos dados, sendo que as notícias do conjunto de dados ficaram dividas em 6 categorias como mostra a tabela 4.

Tabela 4 – Categorias do corpus *Fake.br*

Categorias	Quantidade	%
Politica	4180	58.0
TV & celebridades	1544	21.4
Sociedade & notícias diárias	1276	17.7
Ciência e Tecnologia	112	1.5
Economia	44	0.7
Religião	44	0.7
Total	7200	100

Fonte: (MONTEIRO *et al.*, 2018)

Com o pré-processamento desses dados os modelos deveram ser capazes de ser pré-treinados e usados para a classificação automática de *fake news*. Nas próximas subseções será abordado métodos para resolver o problema.

4.2.2 Escolha dos modelos

Durante a análise exploratória dos modelos para a classificação de texto usando aprendizado de máquina e aprendizado profundo, o autor preferiu escolher os seguintes modelos *Naive Bayes*, *Logistic Regressio*, *suport vector machine*, *Long short-term memory*, *Bidirectional long-short term memory* (BI-LSTM), BERT, os principais motivos de tais escolhas foram baseados em técnicas semelhantes já usadas em outros trabalhos como:

- (MIHALCEA; STRAPPARAVA, 2009) usando métodos clássicos para a detecção de noticias falsas, com recuso de analise linguística para fazer a classificação;
- (MARKINES *et al.*, 2009) usando modelo de linguagem pré-treinados para um desempenho melhor;
- (CRUZ *et al.*, 2019) com uso técnicas de transferência de conhecimento em um *dataset* com poucas amostras, usando aprendizado multitarefa;
- Tempo de treinamento para os modelos, de forma que seja viável dentro dos limites do autor;
- Facilidade na implementação usando bibliotecas específicas.

4.2.3 Métodos

Para a implementação dos modelos, a linguagem Python foi escolhida devido à sua diversidade de bibliotecas para algoritmos de aprendizado de máquina e aprendizado profundo.

As bibliotecas usadas para a criação dos modelos foram sklearn, torch, transformers, numpy e pandas. sklearn, torch, transformers e keras, que disponibilizam diversos algoritmos modulares prontos e já fazem a paralelização automática das operações de matrizes na em GPU através da linguagem CUDA e para o pré-processamento dos dados foi feito apenas a mudança no tamanho das sentenças e na quantidade de palavras. As demais bibliotecas foram usadas para manipulação dos dados e visualização. Todos esses experimentos foram analisados e testados no google colab e kaggle. A seguir será descrito as técnicas usadas para fazer a classificação de texto. Começando por Count Vectorizer e Tf-idf Vectorizer, são duas técnicas equivalentes que são usualmente usadas em algoritmos de aprendizagem de máquina. Essas técnicas servem para transformar os textos em vetores de características que são aplicados como entrada nos algoritmos, que por sua vez encontra padrões nesses vetores que ajudam na classificação de texto. A seguir será descrito como cada uma se comporta!

- *Count Vectorizer* é uma ferramenta fornecida pela biblioteca scikit-learn em Python. Que é utilizada para converter um determinado texto em um vetor com base na frequência (contagem) de cada palavra que ocorre ao longo do texto. Tornado util quando queremos converter cada palavra em cada texto em vetor de características. *Count Vectorizer* cria uma matriz na qual cada palavra exclusiva é representada por uma coluna da matriz e cada amostra de texto do documento é uma linha na matriz. O valor de cada célula nada mais é do que a contagem da palavra. Como está apresentada na Figura 12

Figura 12 – Frequência das palavras

	Document	abreu	agricultura	cargo	citado	colocar	convida	cupula	daniel	deputado	dilma	expulsao	katia
0	corpus[0]	2	0	0	0	1	0	1	0	0	0	2	2
1	corpus[1]	2	0	0	0	0	1	0	0	0	2	0	2
2	corpus[2]	1	1	1	1	0	0	0	1	1	0	0	1

Fonte: Autor

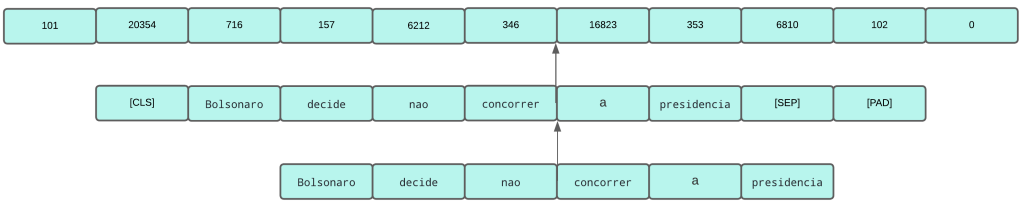
- *Tf-idf Vectorizer* é uma ferramenta fornecida pela biblioteca scikit-learn em Python. É uma técnica que combina dois conceitos, frequência de termo e frequência de documento. Frequência de termo (TF) é o número de ocorrência de um termo no documento, ela indica a importância do termo. A frequência do termo representa cada texto dos dados como uma matriz cujas linhas são o número de documentos e as colunas são o número de termos distintos em todos os documentos. Frequência de documentos(DF) é o número de documentos que contêm um termo específico. A frequência do documento indica quão comum é o termo. A frequência inversa de documentos (IDF) é o peso de um termo, visa

reduzir o peso de um termo se as ocorrências do termo estiverem espalhadas por todos os documentos.

Essas técnicas que foram explicadas foram usadas para classificar as sentenças de texto em *fake news* ou não *fake news*, usando modelos de aprendizagem de máquina. Agora entraremos nas técnicas de aprendizagem profundo, primeiro começamos pelo *tokenizer* que é responsável pelo mapeamento das palavras para inteiros, nos quais as palavras são ordenadas pela frequência. No entanto, é necessário fazer uma diferenciação, pois, os modelos usam dois *tokenizer* diferentes. Para a implementação da LSTM e *BI-LSTM* temos um tokenizdor da seguinte forma, ele recebe como argumento 10000 *num_words*. Isso significa que as 10000 palavras mais frequentes nos textos serão consideras. Depois que o *tokenizer* é ajustado com os dados de treinamento, ele é usado na conversão das palavras para números, na qual é armazenado em *sequences*.

O segundo *tokenizer* é responsável por ajustar os dados de treinamento para o modelo BERT, esse segundo *tokenizer* herda de *PreTrainedTokenizer*, que contém métodos necessários para transformar as sequências em dados de entrada para o BERT, esse método usa um modelo de linguagem pré-treinado para a sumarização. Que é responsável, por trazer os pesos de inicialização nas sequências de textos, no idioma do *dataset*. Essa configuração, é encontrada facilmente no *BertTokenizer* uma classe vindo do *Hugging Face*, que cuida das transformações necessárias do texto de entrada para que ele esteja pronto para ser usado como entrada para modelo BERT. Ele adiciona *tokens* [CLS], [SEP] e [PAD] automaticamente. Como mencionado acima *BertTokenizer* utiliza um modelo de linguagem pré-treinado, o modelo usado para a tarefa foi *neuralmind/bert-base-portuguese-cased*(SOUZA *et al.*, 2019), também do *Hugging Face* no idioma português, esse modelo de linguagem funciona bem em *dataset* em português. Como o problema da pesquisa sugere um *dataset* de notícias em português, nada mais intuitivo que usar um modelo do idioma! Logo abaixo na figura 13 temos um exemplo de como funciona a tokenização em um moledo de linguagem pré-treinado.

Figura 13 – Exemplo de tokenização



Fonte: Autor

Após às duas exemplificações de tokenização, é necessária uma padronização do tamanho das sequências, pois, os modelos exigem que todas as sequências das amostras estejam no mesmo comprimento. Dessa forma, um comprimento é definido como padrão para todas as sequências. Esse tamanho padrão é 500, pois, é um número razoável depois de testes com outros tamanhos, esse foi o que mais se adequou. O preenchimento e truncamento foram configurados através dos parâmetros *padding* e *truncation*, essas configurações são aplicadas na parte final de todas as sequências, se as sequências forem maiores que o tamanho padrão de 500, foram truncadas e sequências menores foram preenchidas com 0, até atingir o comprimento padrão. Mais detalhes no apêndice. A seguir entraremos na exemplificação das modelagens dos modelos escolhidos.

4.2.4 Modelagem

Aqui traremos as abordagens de cada modelo, que foram subsentidos aos experimentos do autor, ao todo são 6 experimentos em que as implementações seguem ideias semelhantes. Esta subseção apresenta detalhes das implementações. Dito isso, começaremos pelos modelos clássicos de aprendizagem de máquina, e depois seguindo a ordem os modelos de aprendizado profundo.

4.2.4.1 Definição da modelagem dos dados

Antes de entrar na modelagem dos modelos, é necessário definir o problema como um problema de classificação binaria, onde 0 (Verdadeiro) é para notícias verdadeiras e 1 (Falso) para notícias falsas.

4.2.4.2 Multinomial Naive Bayes

O algoritmo *Multinomial Naive Bayes* é uma das duas variações do *Naive Bayes* para classificação de texto. A distribuição é parametrizada por vetores $\theta_y = (\theta_{y1}, \ldots, \theta_{yn})$ para cada classe y, onde o número de *features* é n, e θ_{yi} é a probabilidade, $P(x_i|y)$ é a *feature i* que aparece em uma amostra pertencente à classe y. Os parâmetros são estimados por uma versão suavizada de máxima verossimilhança, ou seja, contagem de frequência relativa:

$$\hat{\theta}_{iy} = \frac{N_{yi} + \alpha}{N + \alpha n} \tag{4.1}$$

Onde $N_{yi} = \sum_{x \in T} x_i$ é o número de vezes que a *feature i* aparece em uma amostra de classe y no conjunto de treinamento T, e $N_y = \sum_{i=1}^{n} N_{yi}$ é a contagem total de todas as *features* para a classe y As prioridades de suavização $\alpha \geq 0$ consideram as características não presentes nas amostras de aprendizado e evitam probabilidades zero em cálculos posteriores. A configuração $\alpha = 1$ é chamada de suavização de Laplace, enquanto $\alpha < 1$ é chamada de suavização de Lidstone (PEDREGOSA *et al.*, 2011).

4.2.4.3 Logistic Regresssion

É um algoritmo que pode ser usado para tarefas de regressão e classificação, mas é amplamente usado para tarefas de classificação. A variável de resposta que é binária pertence a uma das classes. É usado para prever variáveis categóricas com a ajuda de variáveis dependentes. Considere que existem duas classes e um novo ponto de dados deve ser verificado a qual classe ele pertence. Em seguida, os algoritmos calculam valores de probabilidade que variam de 0 a 1. Por exemplo, se é uma *fake news* ou não. Na regressão logística, a soma ponderada de entrada é passada pela função de ativação sigmoide e a curva obtida é chamada de curva sigmoide..

A função logística que é uma função sigmoide é uma curva em forma de 'S' que pega quaisquer valores reais e os converte entre 0 e 1. Se a saída dada por uma função sigmoide for maior que $0,5$, a saída é classificada como 1 e se for menor que $0,5$, a saída é classificada como 0. Se o gráfico for para um final negativo, y previsto será 0 e vice-versa (PEDREGOSA *et al.*, 2011).

4.2.4.4 Support vector machine

O classificador baseado em SVM é chamado de *Support Vector Classifier* (SVC) e podemos usá-lo em problemas de classificação. Ele usa o parâmetro de regularização C para otimizar a margem no hiperplano e também é chamado de C-SVC. Para fazer a classificação ele usa dois vetores, um das amostras e outro de características(*features*). Formalmente descrevendo seria da seguinte forma: dados os vetores de treinamento, $x_i \in \mathbb{R}^p$, em duas classes, e um vetor $y \in \{1,-1\}^n$, o objetivo é encontrar $w \in \mathbb{R}^p$ e $b \in \mathbb{R}$ tal que a previsão dada por $\text{sign}(w^T \phi(x) + b)$ seja correta para a maioria das amostras. O SVC resolve o seguinte problema primário:

$$\begin{aligned} &\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i \\ &\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \\ &\zeta_i \geq 0, i = 1, ..., n \end{aligned} \tag{4.2}$$

Intuitivamente, estamos tentando maximizar a margem (minimizando $||w||^2 = w^T w$), enquanto incorremos em uma penalidade quando uma amostra é classificada incorretamente ou dentro do limite da margem. Idealmente, o valor seria $y_i(w^T \phi(x_i) + b)$ para todas as amostras ≥ 1, o que indica uma previsão perfeita. No entanto, os problemas geralmente nem sempre são perfeitamente separáveis com um hiperplano, então permitimos que algumas amostras estejam distantes de seu limite de margem correto. O termo de penalidade C controla a força dessa penalidade e, como resultado, atua como um parâmetro de regularização inverso. Para isso temos um problema dual para o primal que é:

$$\begin{aligned} &\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \\ &\text{subject to } y^T \alpha = 0 \\ &0 \leq \alpha_i \leq C, i = 1, ..., n \end{aligned} \tag{4.3}$$

Onde é o vetor de todos os 1, e Q é uma matriz *nxn* semi-definida por positiva,$Q_{ij} \equiv y_i y_j K(x_i, x_j)$, onde $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ é o kernel. Os termos α_i são chamados de coeficientes duais e são limitados superiormente por C. Essa representação dupla destaca o fato de que os vetores de treinamento são mapeados implicitamente em um espaço dimensional superior (talvez infinito) pela função ϕ.

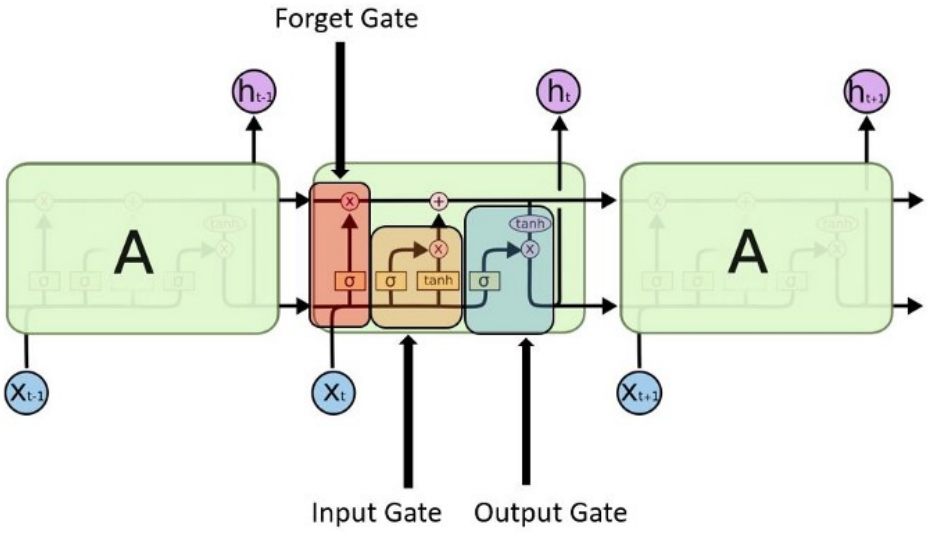
Uma vez que o problema de otimização é resolvido, a saída da função decisão para uma determinada amostra se torna: $\sum_{i \in SV} y_i \alpha_i K(x_i, x) + b$, e a classe prevista corresponde ao seu sinal. Só precisamos somar os vetores de suporte (ou seja, as amostras que estão dentro da margem) porque os coeficientes duais são zero para as outras amostras (PEDREGOSA *et al.*, 2011).

4.2.4.5 LSTM(Long Short Term Memory)

LSTM um algoritmo de aprendizado profundo decorrido de uma rede neural para a classificação automática de texto que faz parte de Redes Neurais Recorrentes, que são capazes de aprender dependências de longo prazo. Uma rede LSTM contém três *gates*. *Forget gate* que

determina aquilo que deve ser esquecido de iterações passadas para essa iteração, *Update gate* que decide o que é importante dessa iteração e deve ser passado adiante e *Output gate* que dá a saída da iteração(HOCHREITER; SCHMIDHUBER, 1997). Como mostra a figura 14

Figura 14 – Modelo LSTM.



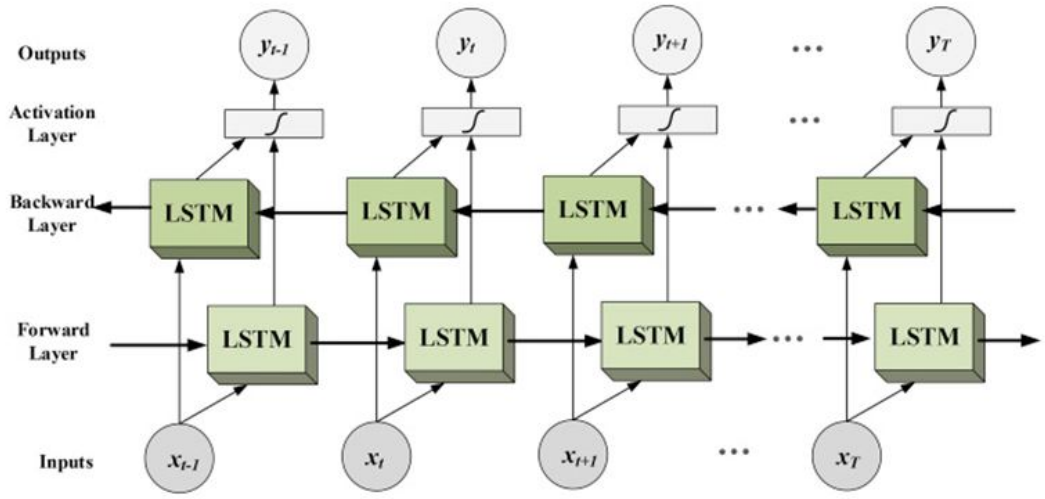
Fonte: (GUDIKANDULA, 2019)

4.2.4.6 BI-LSTM(Bidirectional long-short term memory)

BI-LSTM é o processo de fazer com que qualquer rede neural tenha as informações de sequência em ambas as direções para trás (futuro para passado) ou para frente (passado para futuro) (I2TUTORIALS, 2019).

Em bidirecional, a entrada flui em duas direções, tornando uma BI-LSTM diferente do LSTM regular. Com o LSTM regular, podemos fazer o fluxo de entrada em uma direção, para trás ou para frente. No entanto, em bidirecional, podemos fazer o fluxo de entrada em ambas as direções para preservar as informações futuras e passadas. A memória bidirecional de longo prazo (I2TUTORIALS, 2019).

Figura 15 – Modelo BI-LSTM.



Fonte: (I2TUTORIALS, 2019)

Na imagem acima, podemos ver o fluxo de informações das camadas anteriores e posteriores. O BI-LSTM geralmente é empregado onde a sequência para sequenciar tarefas é necessária. Este tipo de rede pode ser utilizado em modelos de classificação de texto.

4.2.4.7 BERT

BERT é um algoritmo de aprendizado profundo que é baseado em linguagem pré-treinado usando transformador que foi projetado para pré-treinar “representações bidirecionais profundas”, que podem ser ajustadas para diferentes tarefas, com resultados alcançados em *benchmarks* de compreensão em vários idiomas. Como com todos os *transformers*, ele obtém energia de um mecanismo chamado "Atenção", que permite que o modelo calcule a importância ponderada para cada *token* em uma sequência, identificando efetivamente o contexto. A atenção permite que o *transformers* se refira a várias posições em uma sequência para contexto em qualquer momento, independentemente da distância, o que é uma vantagem sobre Redes Neurais Recorrentes (RNR). A vantagem do BERT é sua bidirecionalidade, aproveitando o contexto esquerdo e direito usando um pré-treinamento método denominado “Modelagem de linguagem mascarada”. Além disso, o BERT também se beneficia de ser mais profundo, permitindo capturar mais contexto e informações. BERT-Base, o menor modelo de BERT, possui 12 camadas (768 unidades em cada camada oculta) e 12 cabeças de atenção para um total de 110M de parâmetros. Seu irmão maior, BERT-Large, tem 24 camadas (1024 unidades em cada camada oculta) e 16 cabeças de atenção para um total de 340M de parâmetros(CRUZ *et al.*, 2019).

4.3 Treinamento e Validação

O treinamento dos modelos foi realizado com 80% dos dados da base, 5760 de textos, sendo 2880 com conteúdo verdadeiro e 2880 com conteúdo *fake news*. O treinamento foi realizado com as técnicas anteriores citadas, passando com entradas para os modelos até encontrar um com o melhor resultado de treinamento, para isso uso a técnica de *cross-validation* conhecida como *K-Fold*, onde é método de validação cruzada chamada de *K-Fold* que consiste em dividir o conjunto de dados total em k subconjunto mutuamente exclusivos de mesmo tamanho, a partir daí, um subconjunto é usado para teste e os $k-1$ restantes são usados para estimativa de parâmetros. Quando obtemos um modelo que representa as melhores características, ele salva para que possamos fazer a validação com ele. Essa técnica é viável para os 6 modelos, para o BERT é uma técnica computacionalmente custosa, porquê o BERT têm uma quantidade grande de parâmetros, até mesmo para o BERTbase, então para isso usamos uma variedade nos parâmetros como na curva de aprendizado que variou de $1e-5, 2e-5, 3e-5, 4e-5$ à $5e-5$, outras variações foram no tamanho da sequência que foi testado com $300, 400$ e 500 e época que variou de $3, 4$ e 5 sendo que a melhor representação ficou em $5e-5$ para curva de aprendizado é tamanho da sequência 500 e época 3, e também pela limitação do recurso usado para fazer os experimentos e analise, por ser uma versão *free*, quando ultrapassamos um limite de horas. Ela para a execução e perdemos os dados do experimento. No entanto, quando muda para o *kaggle* percebe-se um ganho, vindo até ser possível executar com *K-Fold* e obter ótimos resultados. O processo de treinamento durou um total de 7h, onde 1h foi para os modelos de aprendizagem de máquina, 2h para os modelos LSTM e 4h para o BERT. Isso no ambiente do *google colab*.

O teste de validação foi os 20% do resto da base de dados, ao final do treinamento dos modelos, eles eram submetidos ao teste de validação para obtemos a análise do desempenho dos modelos na classificação de texto. O treinamento e o teste de validação foi somente usado o corpo textual das notícias.

4.4 Considerações finais do capítulo

Este capítulo apresenta técnicas e conceitos para construir modelos para a classificação de notícias falsas proposto. É importante observar muitos outros conceitos e diferentes modelos foram estudados para ilustrar este trabalho de modo a escolher os classificadores, mas apenas conceitos importantes para que entendemos como é a classificação em cada um deles.

5 RESULTADOS

Após a contextualização do problema e dos principais conceitos para resolução dos problemas aprensados nas seções e capítulos anteriores, será apresentado os resultados computacionais obtidos pelos modelos utilizados. Para isso serão usadas métricas de avaliação para mensurar o desempenho dos modelos, essas métricas são acerto, precisão, cobertura e *f-measure*. Também será descrito o cenário no qual os experimentos serão realizados. A comparação entre os modelos propostos. É análise de algumas técnicas de normalização e por fim a análise que visa verificar os acertos dos modelos propostos. A seguir será descrito as métricas para a análise dos modelos.

5.1 Métricas

A acurácia (ou *accuracy*), diz quantos exemplos foram de fato classificados corretamente, independente da classe. Por exemplo, houver têm 100 observações e 90 delas foram classificadas corretamente, o modelo possui uma acurácia de 90%(KUNUMI, 2020). A acurácia é definida pela fórmula abaixo:

$$Acuracia = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

A precisão (ou *precision*), é uma das métricas mais comuns para avaliar modelos de classificação. Esta métrica é definida pela razão entre a quantidade de exemplos classificados corretamente como positivos e o total de exemplos classificados como positivos(KUNUMI, 2020), de acordo com a fórmula abaixo:

$$Precisao = \frac{TP}{TP + FP} \tag{5.2}$$

A revocação (ou *recall*), é também conhecida como sensibilidade ou taxa de verdadeiro positivo (TPR), dá maior ênfase para os erros por falso negativo. Esta métrica é definida pela razão entre a quantidade de exemplos classificados corretamente como positivos e a quantidade de exemplos que são de fato positivos(KUNUMI, 2020), conforme a fórmula:

$$Renovacao = \frac{TP}{TP + FN} \tag{5.3}$$

f-mensure, leva em consideração a precisão e a revocação. Ela é definida pela média harmônica entre as duas(KUNUMI, 2020), conforme a formula abaixo:

$$F - mensure = 2 * \frac{Precisao * Renovacao}{Precisao + Renovacao} \tag{5.4}$$

5.2 Resultados dos modelos de aprendizado de máquina

Nesta seção serão apresentados os resultados dos modelos de aprendizado de máquina.

5.2.1 Naive Bayes

Após uma busca exaustiva para encontra o melhor modelo, chegou-se aos resultados obtidos na Tabela 5.

Tabela 5 – Treinamento do modelo Naive Bayes com cros-validation e as métricas de avaliação

Folds	**Métricas**			
	accuracy	f1-score	precision	recall
2	0.746	0.668	0.965	0.511
3	0.793	0.748	0.954	0.615
4	0.811	0.777	0.950	0.657
5	0.819	0.788	0.949	0.674
6	0.823	0.795	0.949	0.684
7	0.827	0.800	0.947	0.693
8	0.828	0.801	0.947	0.694
9	0.829	0.803	0.944	0.699
10	**0.832**	**0.807**	**0.946**	**0.704**

Fonte: Autor

Perceba que o modelo que representou melhor foi o do *fold* 10

5.2.2 Avaliação do Naive Bayes

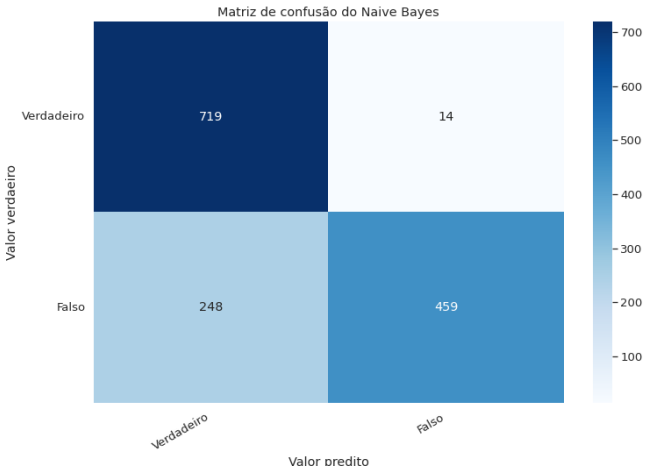
Um modelo com o melhor desempenho de todos nos dados de treinamento foi usado para avaliar no conjunto de teste, obtendo a matriz de confusão da figura 16 e as métricas da Tabela. 6

Tabela 6 – Métricas de avaliação para o teste de validação do Naive Bayes

Classification Report				
	precision	recall	f1-score	support
Verdadeiro	0.77	0.97	0.86	720
Falso	0.96	0.71	0.81	720
accuracy			0.84	1440
macro avg	0.86	0.84	0.84	1440
weighted avg	0.86	0.84	0.84	1440

Fonte: Autor

Figura 16 – Matriz de confusão dos dados de teste



Fonte: Autor

5.2.3 Logistic Regression

Segue o padrão da subseção 5.2.1. O *fold* que melhor represento também foi o 10

Tabela 7 – Treinamento do modelo Logistic Regression com cros-validation e as métricas de avaliação

Folds	**Métricas**			
	accuracy	f1-score	precision	recall
2	0.961	0.962	0.947	0.976
3	0.961	0.961	0.950	0.973
4	0.961	0.961	0.948	0.975
5	0.960	0.961	0.948	0.974
6	0.960	0.961	0.948	0.974
7	0.960	0.961	0.949	0.973
8	0.961	0.962	0.949	0.975
9	0.961	0.962	0.948	0.975
10	**0.962**	**0.962**	**0.951**	**0.975**

Fonte: Autor

5.2.4 Avaliação da Logistic Regression

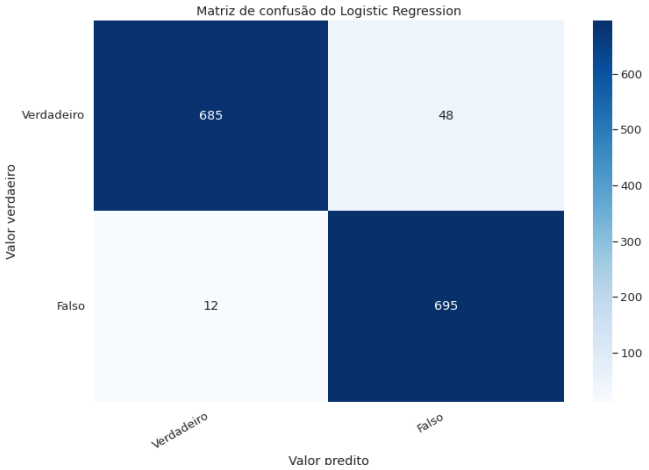
5.2.2 matriz de confusão da Figura 17 e as métricas da Tabela 8 esse despenho todo vem do *fold* 10 que é o melhor entre todos os *folds*.

Tabela 8 – Métricas de avaliação para os dados de teste no Logistic Regression

Classification Report				
	precision	recall	f1-score	support
Verdadeiro	0.98	0.96	0.97	720
Falso	0.96	0.98	0.97	720
accuracy			0.97	1440
macro avg	0.97	0.97	0.97	1440
weighted avg	0.97	0.97	0.97	1440

Fonte: Autor

Figura 17 – Matriz de confusão dos dados de teste



Fonte: Autor

5.2.5 Support Vector Machine

Depois do treinamento com *cross validation*, podemos encontrar o modelo que alcançou a melhor performance, que no caso é modelo do *fold* 2 como mostra na Tabela 9.

Tabela 9 – Treinamento do modelo Support Vector Machine Classifer com cros-validation e as métricas de avaliação

Folds	**Métricas**			
	accuracy	f1-score	precision	recall
2	**0.960**	**0.960**	**0.946**	**0.975**
3	0.957	0.958	0.945	0.972
4	0.959	0.960	0.947	0.973
5	0.957	0.958	0.945	0.970
6	0.958	0.959	0.946	0.972
7	0.959	0.959	0.948	0.971
8	0.959	0.959	0.949	0.971
9	0.957	0.958	0.947	0.970
10	0.957	0.958	0.946	0.971

Fonte: Autor

5.2.6 Avaliação do Support Vector Machine

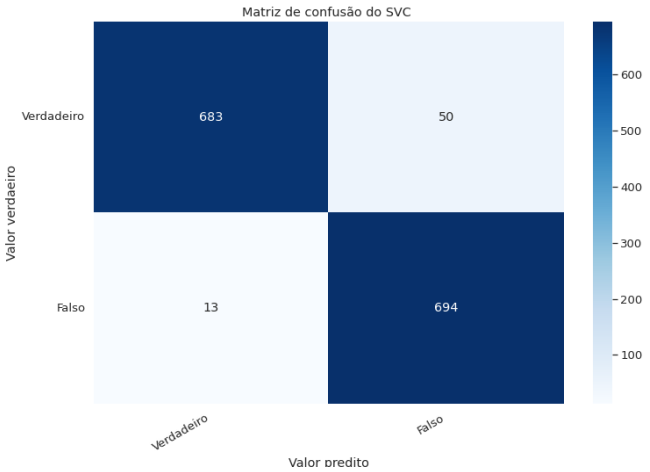
5.2.2 neste caso a matriz de confusão da figura 18 e as métricas da Tabela 10

Tabela 10 – Métricas de avaliação para os dados de teste no Support Vector Machine Classifier

Classification Report				
	precision	recall	f1-score	support
Verdadeiro	0.98	0.95	0.96	720
Falso	0.96	0.98	0.97	720
accuracy			0.97	1440
macro avg	0.97	0.97	0.97	1440
weighted avg	0.97	0.97	0.97	1440

Fonte: Autor

Figura 18 – Matriz de confusão dos dados de teste



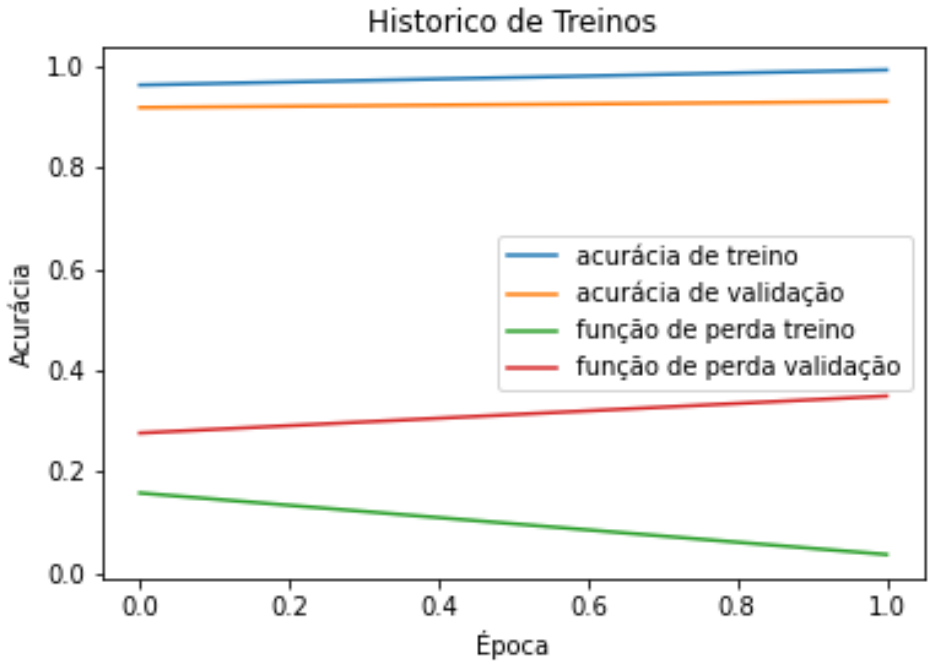
Fonte: Autor

5.3 Resultados dos modelos de aprendizado profundo

5.3.1 LSTM

Modelo treinado com 7 épocas, e com uma *callback* para monitoramento a função de perda. É necessário, pois, evita que o modelo entre em *overfitting*. Para extrair um modelo que melhor represente, foi usado o *cross validation* com 5 *fold*, a figura 19 , mostra como o modelo se saiu. Na validação cruzada, o modelo obteve uma acurácia de 98% no treinamento.

Figura 19 – Acurácia e função de perda do modelo em relação as épocas de treinamento, em azul valores para o conjunto de treinamento e em laranja para o conjunto de validação, em vermelho a função de perda da validação e em verde a de treino



Fonte: Autor

5.3.2 Avaliação da LSTM

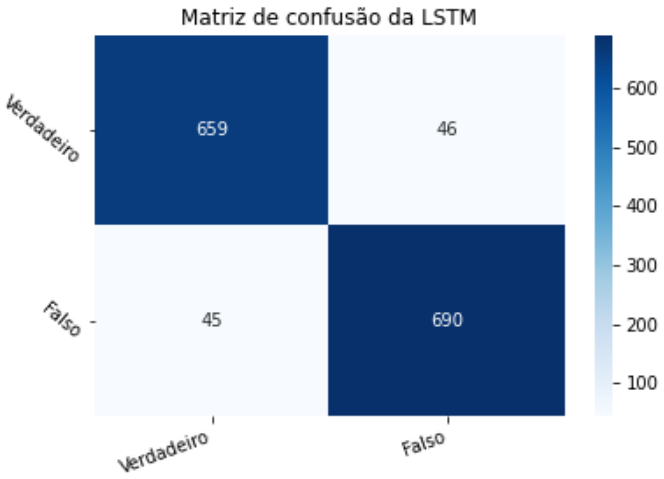
Após a validação cruzada para extrair o melhor modelo, ele foi usado para avaliar no conjunto de teste, obtendo a matriz de confusão da figura. 20 e as métricas da Tabela 11

Tabela 11 – Métricas de avaliação para os dados de teste na LSTM simples

Classification Report				
	precision	recall	f1-score	support
Verdadeiro	0.96	0.94	0.95	717
Falso	0.94	0.96	0.95	723
accuracy			0.95	1440
macro avg	0.95	0.95	0.95	1440
weighted avg	0.95	0.95	0.95	1440

Fonte: Autor

Figura 20 – Matriz de confusão dos dados de teste

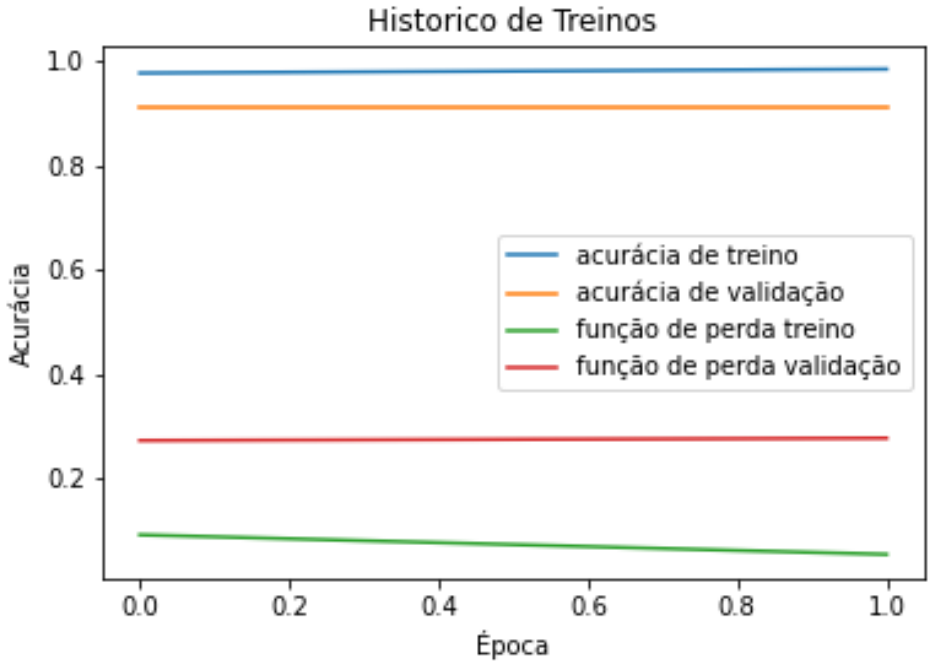


Fonte: Autor

5.3.3 BI-LSTM

5.3.1 com as mesmas configurações e mesma acurácia de treinamento, como é demostrado na figura 21.

Figura 21 – Acurácia e função de perda do modelo em relação as épocas de treinamento, em azul valores para o conjunto de treinamento e em laranja para o conjunto de validação, em vermelho a função de perda da validação e em verde a de treino



Fonte: Autor

5.3.4 Avaliação da BI-LSTM

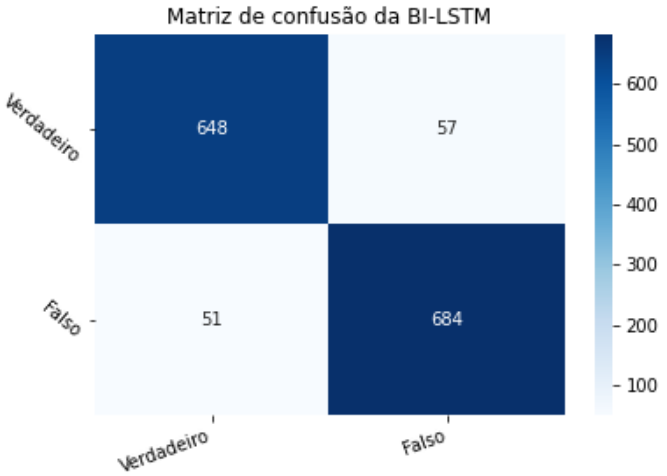
5.3.2 neste caso, o modelo teve uma acurácia inferior ao outro modelo. Obtendo a matriz de confusão da figura 22 e as métricas da Tabela 12

Tabela 12 – Métricas de avaliação para os dados de teste na LSTM bidirecional

Classification Report				
	precision	recall	f1-score	support
Verdadeiro	0.95	0.93	0.94	717
Falso	0.93	0.95	0.94	723
accuracy			0.94	1440
macro avg	0.94	0.94	0.94	1440
weighted avg	0.94	0.94	0.94	1440

Fonte: Autor

Figura 22 – Matriz de confusão dos dados de teste

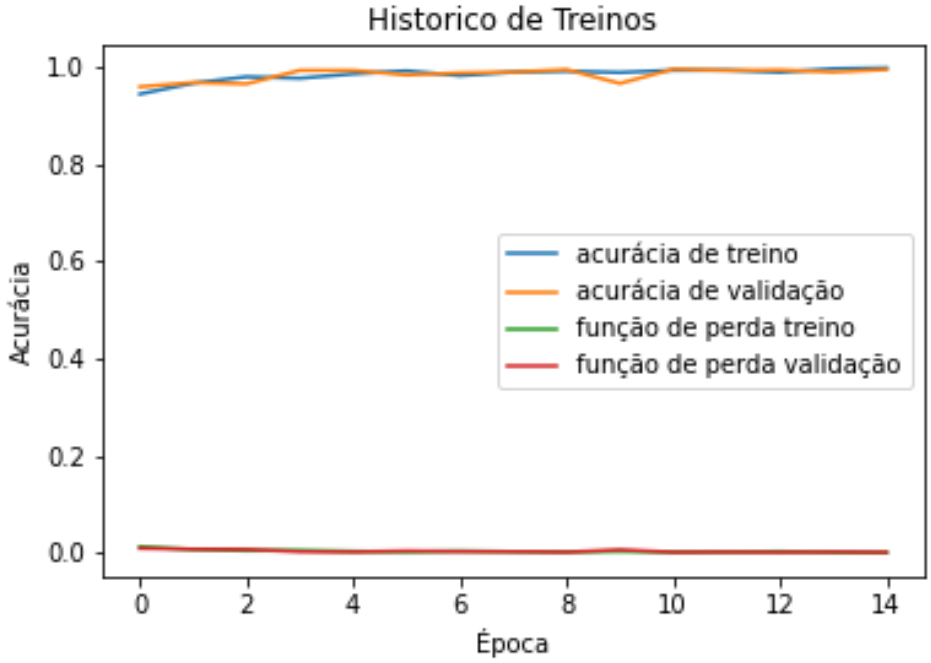


Fonte: Autor

5.3.5 BERT

No treinamento do BERT, foi utilizado a *cross validation* para extrair o melhor modelo. As configurações para obter o melhor modelo foram 3 épocas de treinamento com a variação na *learning rate*, com 5 *folds*. Com isso obteve uma acurácia de 99% como mostra a figura 23.

Figura 23 – Acurácia e função de perda, em relação a validação cruzada no treinamento, em azul a acurácia de treino e laranja a acurácia de validação, e as retas de baixo são referente a função de perda, onde a reta verde é *loss* de treinamento, enquanto que vermelho é a *loss* validação



Fonte: Autor

5.3.6 Avaliação do BERT.

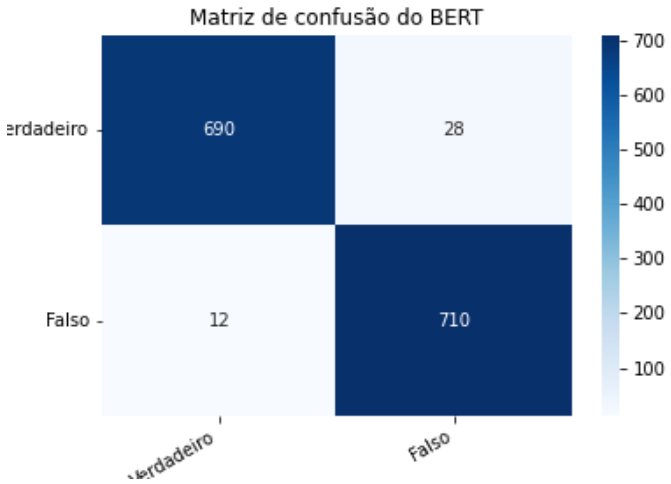
Após obter um modelo com as melhores configurações, ele foi usado para avaliar os dados de teste, obtendo a matriz de confusão da figura 24 e as métricas da Tabela 13

Tabela 13 – Métricas de avaliação para os dados de teste no BERT

Classification Report				
	precision	recall	f1-score	support
Verdadeiro	0.97	0.98	0.97	720
Falso	0.96	0.98	0.97	720
accuracy			0.97	1440
macro avg	0.97	0.97	0.97	1440
weighted avg	0.97	0.97	0.97	1440

Fonte: Autor

Figura 24 – Matriz de confusão dos dados de teste

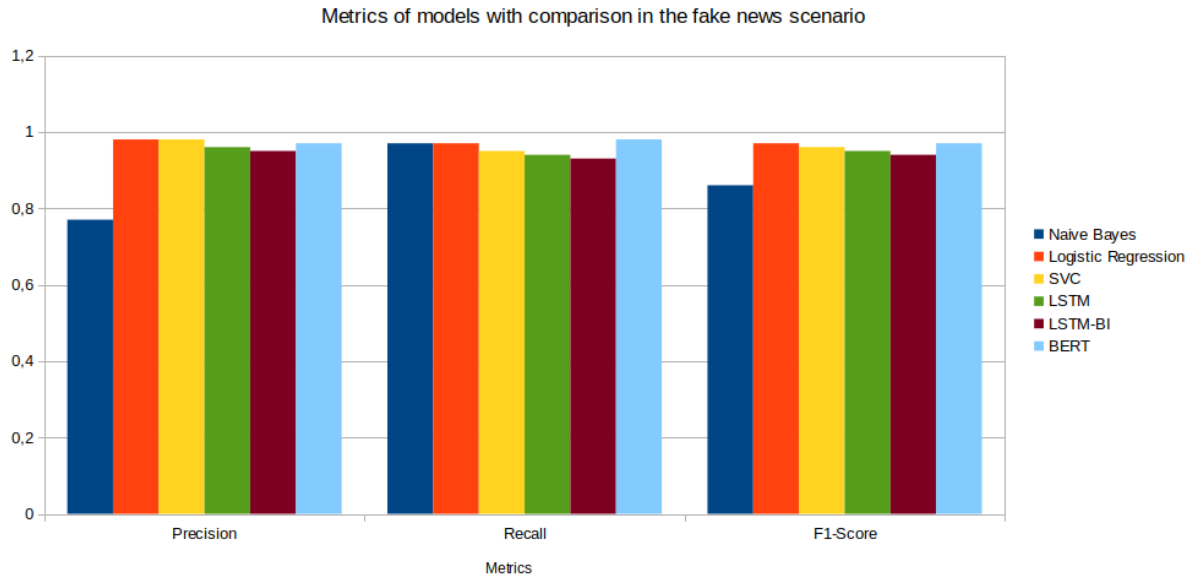


Fonte: Autor

5.4 Comparação dos experimentos

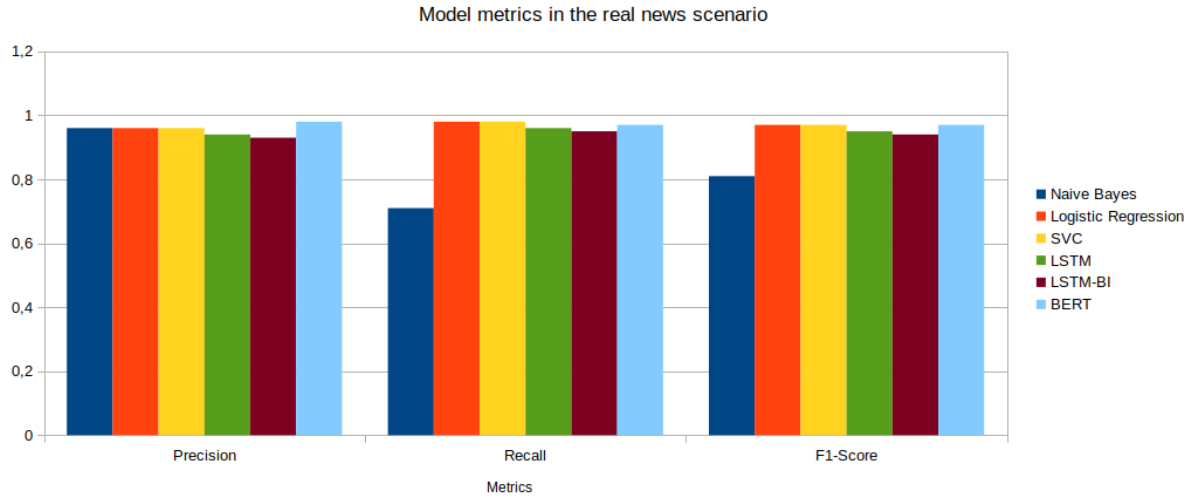
A primeira figura 25 representa o cenário das *fake news*, enquanto a segunda figura 26 representa o cenário das notícias reais. Como pode ser visto, o modelo que mais se destaca é o BERT em todas as métricas e nos dois cenários, os outros modelos foram inferiores em relação as métricas de avaliação.

Figura 25 – Desempenho dos modelos no cenário das *fake news*, aqui temos a comparação dos modelos em relação as métricas de avaliação



Fonte: Autor

Figura 26 – Desempenho dos modelos no cenário das notícias reais, aqui temos a comparação dos modelos em relação as métricas de avaliação



Fonte: Autor

O classificador final construído atende às expectativas do projeto e realiza resultados comparáveis com outros modelos de últimas gerações sobre o problema das *fake news*. A escolha dos modelos utilizados e suas implementações são realizadas para obter o melhor classificador. Olhando para os resultados, fica claro que são contribuições relevantes, porque poucos trabalhos foram encontrados no âmbito das *fake news* em português brasileiro, e com ótimos desempenhos em todos os cenários de notícias verdadeiras ou falsas. No entanto, a parte mais importante deste trabalho é fazer discussões sobre esse tema foram levantadas no Brasil e no mundo. este problema vem afetando muito o Brasil, causando perseguição, morte, concorrência desleal, crise econômica e assim por diante. O Brasil é um pouco distante em relação a outros países para resolver tais problemas.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve como objetivo construir um classificador de aprendizado profundo com transferência de conhecimento capaz de classificar notícias falsas em português brasileiro. Para isso, o modelo deve ser construído para utilizar apenas o corpo da notícia, pré-processando com uma pequena quantidade de dados. Para esta tarefa, o modelo BERT é usado para construir o classificador. Os resultados da validação para o classificador proposto são apresentados no capítulo anterior. Os resultados obtidos são satisfatórios e em consonância com os objetivos propostos, o classificador atinge alto desempenho na base de dados selecionada e requer pouco pré-processamento dos dados.

Observando os resultados, percebe-se que o modelo principal responde bem os objetivos, o BERT tem um ótimo desempenho quando se trata de tarefas classificação de texto, pois usa transferência de conhecimento para novas tarefas,e com o uso do BERTimbau como modelo pré-treinado melhora muito no tempo e ajuste da tarefa, pois o modelo pré-treinado é do idioma português vindo a melhorar na adaptação de domínio. O que facilitou o ajuste de características tais como mudar a quantidade de épocas, curvas de aprendizado e tamanho das sequências.Portanto, é notável que são contribuições relevantes visto que foram encontrados poucos trabalhos no âmbito das *fake news* no português brasileiro e com desempenho similar ao deste trabalho. Todavia é importante ressaltar que este trabalho faz o levantamento da discussão sobre esse tema que persegue o Brasil e o mundo. Esse problema vem afetando o Brasil drasticamente, causando perseguições, mortes, concorrência desleal, crise econômica, entre outros. Longe dos outros países, o Brasil se encontra um pouco longe de resolver tal problema.

Contundo, mesmo com poucos trabalhos, já começamos a combater. Acredita-se que em um momento no futuro existirão modelos como esses que farão parte de uma gama de trabalhos no âmbito das *fake news*. Que conseguiram uma alta performance na identificação das notícias falsas, é partindo dessas pesquisas possa gerar uma ferramenta, que seja capaz de generalizar na identificação das notícias falsas e se popularizar na comunidade leiga, causando um impacto positivo na sociedade promovendo assim a diminuição da desinformação.

Na elaboração deste projeto foi discutido vários algoritmos em primeiro momento, mas o que parecia não tão distante era o BERT, por ter uma gama de explicações. Durante o processo de aprendizagem o autor errou algumas vezes, por sua insipiência na areá de Aprendizado profundo. Ainda assim, isso não foi empecilho, com o amadurecimento do projeto, os cenários

foram mudando e desenrolando-se. O processo de classificar *fake news* não é intuitivo, pois, o contexto muda a cada momento, mas com técnicas de aprendizado profundo e transferência de conhecimento torna-se mais plausível de classifica-las.

Como trabalho futuro, observa-se a necessidade de um aumento e atualização do *dataset* para melhorar mais no refinamento do modelo, ou partir para a otimização criando um modelo de programação linear de maximização junto com BERT, o modelo de maximização seria para reter a proliferação de noticiais enganosas, o que casa muito bem com a criação de uma ferramenta ou aplicativo de detecção de *fake news* para os usuários leigos de informação. também há uma necessidade criar novos modelos pré-treinados para transformador. É Para futuras pesquisas o código fonte do projeto se encontra no *github* neste link github.com/classificadores.

REFERÊNCIAS

ALLCOTT, H.; GENTZKOW, M. Social media and fake news in the 2016 election. **Journal of Economic Perspectives**, v. 31, n. 2, p. 211–36, May 2017. Disponível em: <http://www.aeaweb.org/articles?id=10.1257/jep.31.2.211>.

AMINE, B. M.; DRIF, A.; GIORDANO, S. Merging deep learning model for fake news detection. In: IEEE. **2019 International Conference on Advanced Electrical Engineering (ICAEE)**. [S.l.], 2019. p. 1–4.

ARAÚJO, F. H.; CARNEIRO, A.; SILVA, R. R.; MEDEIROS, F. N.; USHIZIMA, D. M. Redes neurais convolucionais com tensorflow: Teoria e prática. **SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. III Escola Regional de Informática do Piauí. Livro Anais-Artigos e Minicursos**, Sociedade Brasileira de Computação, v. 1, p. 382–406, 2017.

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.

BAYES, T. de. **Teorema de Bayes — Wikipédia, a enciclopédia livre**. 2022. [Online; accessed 6-julho-2022]. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Teorema_de_Bayes&oldid=63939256>.

BICKEL, S.; SAWADE, C.; SCHEFFER, T. Transfer learning by distribution matching for targeted advertising. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2009. p. 145–152.

BRAGA, A. d. P. **Redes neurais artificiais: teoria e aplicações**. [S.l.]: Livros Técnicos e Científicos, 2000.

BUNTAIN, C.; GOLBECK, J. Automatically identifying fake news in popular twitter threads. In: IEEE. **2017 IEEE International Conference on Smart Cloud (SmartCloud)**. [S.l.], 2017. p. 208–215.

CERI, S.; BOZZON, A.; BRAMBILLA, M.; VALLE, E. D.; FRATERNALI, P.; QUARTERONI, S. An introduction to information retrieval. In: **Web information retrieval**. [S.l.]: Springer, 2013. p. 3–11.

CHUA, L. O.; YANG, L. Cellular neural networks: Theory. **IEEE Transactions on circuits and systems**, IEEE, v. 35, n. 10, p. 1257–1272, 1988.

CRUZ, J. C. B.; TAN, J. A.; CHENG, C. Localization of fake news detection via multitask transfer learning. **arXiv preprint arXiv:1910.09295**, 2019.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805**, 2018.

FOLHA DE S.PAULO. **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: 27 mai. 2020.

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

GUDIKANDULA purnasai. **Recurrent Neural Networks and LSTM explained**. 2019. [Online; accessed 7-August-2022]. Disponível em: <https://purnasaigudikandula.medium.com/recurrent-neural-networks-and-lstm-explained-7f51c7f6bbb9>.

GUYON, I. Neural networks and applications tutorial. **Physics Reports**, Elsevier, v. 207, n. 3-5, p. 215–259, 1991.

HARTMANN, N.; FONSECA, E.; SHULBY, C.; TREVISO, M.; RODRIGUES, J.; ALUISIO, S. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. **arXiv preprint arXiv:1708.06025**, 2017.

HAYKIN, S. **Redes neurais: princípios e prática. 2001**. [S.l.]: Bookman, Porto Alegre, Rio Grande do Sul, Brasil, 2001.

HEBB, D. O. **The organization of behavior: a neuropsychological theory**. [S.l.]: J. Wiley; Chapman & Hall, 1949.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HOREV, R. **BERT Explained: State of the art language model for NLP**. 2018. Disponível em: <https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270>. Acesso em: 22 jul. 2020.

HOWARD, J.; RUDER, S. Universal language model fine-tuning for text classification. **arXiv preprint arXiv:1801.06146**, 2018.

I2TUTORIALS. **Deep Dive into Bidirectional LSTM**. 2019. [Online; accessed 8-August-2022]. Disponível em: <https://purnasaigudikandula.medium.com/recurrent-neural-networks-and-lstm-explained-7f51c7f6bbb9>.

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.

KALIYAR, R. K. Fake news detection using a deep neural network. In: IEEE. **2018 4th International Conference on Computing Communication and Automation (ICCCA)**. [S.l.], 2018. p. 1–7.

KANADE, V. **What Is Logistic Regression? Equation, Assumptions, Types, and Best Practices**. 2022. [Online; accessed 6-August-2022]. Disponível em: <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/>.

KUNUMI. **Métricas de Avaliação em Machine Learning: Classificação**. 2020. Disponível em: <https://medium.com/kunumi/m\%C3\%A9tricas-de-avalia\%C3\%A7\%C3\%A3o-em-machine-learning-classifica\%C3\%A7\%C3\%A3o-49340dcdb198>. Acesso em: 12 nov. 2020.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.

LEE, C. **Understanding Bidirectional RNN in PyTorch**. 2017. [Online; accessed 7-August-2022]. Disponível em: <https://towardsdatascience.com/understanding-bidirectional-rnn-in-pytorch-5bd25a5dd66>.

LIANG, L.; ZHENG, J.; FU, J. Sentence classification with transfer network. In: IEEE. **2018 International Conference on Information Systems and Computer Aided Education (ICISCAE)**. [S.l.], 2018. p. 379–382.

MARKINES, B.; CATTUTO, C.; MENCZER, F. Social spam detection. In: . [S.l.: s.n.], 2009. p. 41–48.

MARSLAND, S. **Machine learning: an algorithmic perspective**. [S.l.]: CRC press, 2015.

MAXIME. **What is a Transformer?** 2019. Disponível em: <https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04>. Acesso em: 8 jul. 2020.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943.

MIHALCEA, R.; STRAPPARAVA, C. The lie detector: Explorations in the automatic recognition of deceptive language. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the ACL-IJCNLP 2009 Conference Short Papers**. [S.l.], 2009. p. 309–312.

MINSKY, M.; PAPERT, S. A. **Perceptrons: An introduction to computational geometry**. [S.l.]: MIT press, 2017.

MITTAL, S. **Deep Learning Techniques for Text Classification**. 2019. Disponível em: <https://medium.com/datadriveninvestor/deep-learning-techniques-for-text-classification-9392ca9492c7>. Acesso em: 27 out. 2020.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. **Foundations of machine learning**. [S.l.]: MIT press, 2018.

MOKEYLEARN. **Text Classification**. 2019. Disponível em: <https://monkeylearn.com/text-classification/>. Acesso em: 20 out. 2020.

MOLTZAU, A. **What is Transfer Learning?** 2019. Disponível em: <https://medium.com/@alexmoltzau/what-is-transfer-learning-6ebb03be77ee>. Acesso em: 28 nov. 2020.

MONTEIRO, R. A.; SANTOS, R. L.; PARDO, T. A.; ALMEIDA, T. A. de; RUIZ, E. E.; VALE, O. A. Contributions to the study of fake news in portuguese: New corpus and automatic detection results. In: SPRINGER. **International Conference on Computational Processing of the Portuguese Language**. [S.l.], 2018. p. 324–334.

MURPHY, K. P. **Machine learning: a probabilistic perspective**. [S.l.]: MIT press, 2012.

OTTONICAR, S. L. C.; VALENTIM, M.; JORGE, L. F.; MOSCONI, E. Fake news, big data e o risco à democracia: novos desafios à competência em informação e midiática. 2019.

PAN, S. J.; YANG, Q. A survey on transfer learning. **IEEE Transactions on knowledge and data engineering**, IEEE, v. 22, n. 10, p. 1345–1359, 2009.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PÉREZ-ROSAS, V.; KLEINBERG, B.; LEFEVRE, A.; MIHALCEA, R. Automatic detection of fake news. **arXiv preprint arXiv:1708.07104**, 2017.

PETERS, M. E.; NEUMANN, M.; IYYER, M.; GARDNER, M.; CLARK, C.; LEE, K.; ZETTLEMOYER, L. Deep contextualized word representations. **arXiv preprint arXiv:1802.05365**, 2018.

POLITIZE. **Notícias falsas e pós-verdade: o mundo das fake news e da (des)informação**. 2017. Disponível em: <https://www.politize.com.br/noticias-falsas-pos-verdade/>. Acesso em: 27 mai. 2020.

RASCHKA, S. **Python machine learning**. [S.l.]: Packt publishing ltd, 2015.

ROSENBLATT, F. **The perceptron: A theory of statistical separability in cognitive systems**. [S.l.]: United States Department of Commerce, 1958.

SATPAL, S.; SARAWAGI, S. Domain adaptation of conditional probability models via feature subsetting. In: SPRINGER. **European Conference on Principles of Data Mining and Knowledge Discovery**. [S.l.], 2007. p. 224–235.

SCHÖLKOPF, B.; SMOLA, A. J.; BACH, F. *et al.* **Learning with kernels: support vector machines, regularization, optimization, and beyond**. [S.l.]: MIT press, 2002.

SIDOROV, G. **Syntactic n-grams in computational linguistics**. [S.l.]: Springer, 2019.

SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. Portuguese named entity recognition using bert-crf. **arXiv preprint arXiv:1909.10649**, 2019. Disponível em: <http://arxiv.org/abs/1909.10649>.

Support Vector Machine. **Support-vector machine — Wikipedia, The Free Encyclopedia**. 2022. [Online; accessed 4-August-2022]. Disponível em: <https://en.wikipedia.org/w/index.php?title=Support-vector_machine&oldid=1100727517>.

TAN, B.; SONG, Y.; ZHONG, E.; YANG, Q. Transitive transfer learning. In: **Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. [S.l.: s.n.], 2015. p. 1155–1164.

THOTA, A.; TILAK, P.; AHLUWALIA, S.; LOHIA, N. Fake news detection: a deep learning approach. **SMU Data Science Review**, v. 1, n. 3, p. 10, 2018.

USP. **Redes Neurais Artificiais**. 2009. Disponível em: <https://sites.icmc.usp.br/andre/research/neural/>. Acesso em: 28 nov. 2020.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2017. p. 5998–6008.

WARNER, A. J. Natural language processing. **Annual review of information science and technology**, v. 22, p. 79–108, 1987.

YANG, Y.; ZHENG, L.; ZHANG, J.; CUI, Q.; LI, Z.; YU, P. S. **TI-CNN: Convolutional Neural Networks for Fake News Detection**. 2018.

ZHANG, J.; DONG, B.; YU, P. S. Fakedetector: Effective fake news detection with deep diffusive neural network. 2018.