UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

RAFAELA DE AMORIM BARBOSA SILVA

DETECÇÃO AUTOMÁTICA DE FAKE NEWS EM TWEETS SOBRE AS ELEIÇÕES BRASILEIRAS DE 2022

CAMPINA GRANDE - PB

2023

Rafaela de Amorim Barbosa Silva

DETECÇÃO AUTOMÁTICA DE FAKE NEWS EM TWEETS SOBRE AS ELEIÇÕES BRASILEIRAS DE 2022

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador : Eanes Torres Pereira

CAMPINA GRANDE - PB

2023

Rafaela de Amorim Barbosa Silva

DETECÇÃO AUTOMÁTICA DE FAKE NEWS EM TWEETS SOBRE AS ELEIÇÕES BRASILEIRAS DE 2022

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Eanes Torres Pereira

Orientador – UASC/CEEI/UFCG

Carlos Eduardo Santos Pires

Examinador – UASC/CEEI/UFCG

Francisco Vilar Brasileiro

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 28 de Junho de 2023.

CAMPINA GRANDE - PB

RESUMO

Fake News, ou notícias falsas, são informações deliberadamente falsas ou enganosas criadas e disseminadas com o intuito de enganar o público. Esses artigos de notícia são feitos para se parecerem com notícias legítimas. Seu objetivo é manipular a opinião pública, espalhar desinformação, influenciar eleições, gerar controvérsia ou usados para ganhos financeiros. Com o advento das redes sociais, as pessoas começaram a consumir suas notícias *online*, pois é extremamente simples, rápido e facilmente acessível. No entanto, essa facilidade também levou a um aumento na disseminação das notícias falsas. Nos últimos anos, temos visto que as eleições e a opinião pública sobre causas sociais importantes têm sido influenciadas pelo espalhamento das *fake news*. Elas são criadas e proliferam-se rapidamente, ressaltando a urgência da necessidade para que sua detecção seja rápida. Neste trabalho, é proposta uma metodologia para detecção de *fake news* usando redes neurais profundas, com um conjunto de dados com mais de 2 milhões de *tweets* sobre as eleições presidenciais Brasileiras de 2022, rotuladas automaticamente por um modelo de supervisão fraca, obtivemos F1-score de 98% em *tweets* que não contém *fake news*, e F1-score de 47% em *tweets* contendo *fake news*.

AUTOMATIC FAKE NEWS DETECTION ON TWEETS ABOUT THE BRAZILIAN ELECTIONS OF 2022

ABSTRACT

Fake News are deliberately false or misleading information created and disseminated with the aim of deceiving the public. These news articles are often designed to resemble legitimate news. Their objective is to manipulate public opinion, spread misinformation, influence elections, generate controversy, or to have financial gains. With the advent of social media, people have started consuming news online as it is extremely simple, fast, and easily accessible. However, this has also led to an increase in the dissemination of fake news. In recent years, we have seen that elections and public opinion on important social issues have been influenced by the spread of fake news. They are created and spread rapidly, highlighting the urgent need for rapid detection. In this work, we propose a methodology for detecting fake news using deep neural networks, with a dataset of over 2 million tweets from the Brazilian presidential elections of 2022, labeled automatically by a weak supervision model, with F1-score of 98% on non-fake news tweets, and F1-score of 47% on tweets containing fake news.

Detecção Automática de *fake news* em Tweets sobre as Eleições Brasileiras de 2022

Trabalho de Conclusão de Curso

Rafaela de Amorim Barbosa Silva (Aluna)*
Eanes Torres Pereira (Orientador)†
Universidade Federal de Campina Grande
Departamento de Sistemas e Computação
Campina Grande, Paraíba - Brasil

ABSTRACT

Fake News are deliberately false or misleading information created and disseminated with the aim of deceiving the public. These news articles are often designed to resemble legitimate news. Their objective is to manipulate public opinion, spread misinformation, influence elections, generate controversy, or to have financial gains. With the advent of social media, people have started consuming news online as it is extremely simple, fast, and easily accessible. However, this has also led to an increase in the dissemination of fake news. In recent years, we have seen that elections and public opinion on important social issues have been influenced by the spread of fake news. They are created and spread rapidly, highlighting the urgent need for rapid detection. In this work, we propose a methodology for detecting fake news using deep neural networks, with a dataset of over 2 million *tweets* from the Brazilian presidential elections of 2022, labeled automatically by a weak supervision model, with F1-score of 98% on non-fake news *tweets*, and F1-score of 47% on *tweets* containing fake news.

KEYWORDS

Fake News, Tweets, Automatic Labeling, Weak Supervision, Convolutional Neural Networks, CNN, Fake News Detection

1 INTRODUÇÃO

As notícias falsas consistem na distribuição deliberada de informações irreais que apelam para o emocional de quem consome a notícia[1]. Elas se espalham principalmente através de redes sociais com manchetes sensacionalistas, tendenciosas e exageradas para chamar a atenção a fim de se obter ganhos financeiros ou políticos[2]. Apesar de não ser um fenômeno exclusivo do período eleitoral, é durante esta época que as *fake news* ganham poder e se espalham como nunca, uma prática muito recorrente no Brasil desde a República Velha (1889)[8], e desde meados de 2017, os brasileiros usam o termo *fake news* que se popularizou principalmente por conta das eleições presidenciais que ocorreram no final de 2018. Seguindo os passos de presidenciáveis dos EUA em anos anteriores[3], nesse período foi observado um amplo uso de redes sociais como veículo de propaganda política, bem como veículo de desinformação que beneficiaria algum candidato[4] [2][1].

A eleição presidencial de 2022 foi a disputa mais acirrada no Brasil democrático até então, com a menor diferença percentual da história entre dois candidatos em segundo turno. Com 99,96% de urnas apuradas, o atual presidente Lula ganhou de Bolsonaro com menos de dois pontos percentuais de diferença, um pouco mais de 2 milhões de votos[5] [6]. A incerteza de quem seria o presidente pelos próximos quatro anos e a divisão do Brasil em dois polos políticos, esquerda e direita, ficou bastante evidente através das redes sociais conforme o período eleitoral se aproximava. Nesta pesquisa, serão usadas publicações do Twitter para treinar um modelo de aprendizagem profunda que consiga identificar *fake news* em um texto pequeno

O Twitter é caracterizado por ser uma rede social de *microblogging*, os usuários podem publicar até no máximo 280 caracteres por vez. Em 2022, a rede contava com mais de 19 milhões de usuários brasileiros[7], cerca de 10% da população brasileira total. Há uma grande quantidade de informações que correm todos os dias na rede social, com ampla disseminação incentivada pela própria plataforma conforme alguma publicação "viraliza" [8] [4]. Da mesma maneira, informações falsas são espalhadas mais rapidamente do que notícias verdadeiras, e mais rapidamente do que especialistas conseguem identificar e provar de maneira clara que são irreais. Por exemplo, idosos são o público-alvo preferencial de *fake news* por serem mais propensos a acreditarem e espalharem notícias falsas[6], por exemplo, próximo à data da votação para presidente no segundo turno de 2022, foi espalhado em redes sociais que pessoas aposentadas podiam usar o voto como prova de vida somente se o eleitor votasse em um candidato específico[9] – na verdade, o voto realmente pode ser usado como prova de vida para o INSS mas é independente do voto ter sido 13, 22, branco ou nulo.

*e-mail: rafaela.silva@ccc.ufcg.edu.br
†e-mail: eanes@computacao.ufcg.edu.br

[1]Acesso: https://www.tse.jus.br/comunicacao/noticias/2022/Junho/pilulas-contra-a-desinformacao-desconfie-de-mensagens-com-apelo-emocional
[2]https://reporterbrasil.org.br/2019/06/as-redes-sociais-no-mundo-das-fake-news/
[3]https://www.theguardian.com/world/2012/feb/17/obama-digital-data-machine-facebook-election
[4]https://www1.folha.uol.com.br/poder/2018/11/90-dos-eleitores-de-bolsonaro-acreditaram-em-fake-news-diz-estudo.shtml
[5]https://www.cnnbrasil.com.br/politica/disputa-entre-lula-e-bolsonaro-e-a-eleicao-para-presidente-mais-acirrada-da-historia/
[6]https://www.bloomberglinea.com.br/2022/10/30/como-lula-e-bolsonaro-chegam-a-eleicao-mais-acirrada-da-era-democratica/
[7]https://valorinveste.globo.com/mercados/internacional-e-commodities/noticia/2022/04/25/brasil-tem-a-quarta-maior-base-de-usuarios-do-twitter-no-mundo.ghtml
[8]https://super.abril.com.br/tecnologia/no-twitter-fake-news-se-espalham-6-vezes-mais-rapido-que-noticias-verdadeiras/
[9]https://g1.globo.com/fato-ou-fake/eleicoes/noticia/2022/10/13/e-fake-que-votar-so-serve-como-prova-de-vida-se-eleitor-escolher-uma-candidatura-especifica.ghtml

Existem jornalistas comprometidos em checar a veracidade de notícias duvidosas, como no caso do Projeto Comprova[10], Agência Lupa[11], Fato ou Fake[12], entre outros. Porém, a checagem manual de fatos pode levar um tempo, é preciso uma maneira automatizada para verificação de veracidade das notícias. Para ajudar jornalistas, e a população no geral, nessa tarefa da identificação de notícias falsas, a inteligência artificial pode se mostrar bastante útil, podemos usar modelos de redes neurais para agilizar o trabalho de verificação de notícias.

As redes neurais podem ser usadas em uma ampla variedade de tarefas, como reconhecimento de padrões, classificação, processamento de linguagem natural, visão computacional e muito mais. Elas são capazes de aprender e generalizar a partir de exemplos fornecidos durante o treinamento. Já existem esforços para fazer a detecção automática de informações falsas para ajudar no controle da praga da desinformação, como em [12] e a ferramenta disponível em[13], também existem estudos sob perspectivas causais da influência das *fake news* no debate sobre as eleições presidenciais estadunidenses de 2016 [2]. Ao fim do presente trabalho pretende-se ter um uma rede neural profunda que consiga identificar a presença de *fake news* em um *tweet*, e também, uma metodologia de como lidar com bases de dados grandes não rotuladas através de um modelo de aprendizado que usa funções ruidosas para aprender padrões em um *tweet*.

2 REVISÃO DE LITERATURA

O artigo [12] trata-se sobre detecção de notícias falsas nas redes sociais, fazendo uma revisão abrangente sobre o assunto, sobre algoritmos existentes sob a perspectiva da mineração de dados, métricas de avaliação e conjuntos de dados representativos. As *fake news* apresentam características únicas, ambíguas, são escritas intencionalmente para enganar os leitores e fazer com que acreditem em informações falsas, o que torna difícil e não trivial detectá-las com base no conteúdo da notícia. Além disso, há dois fatores principais que fazem os consumidores suscetíveis a fake news naturalmente, o realismo ingênuo onde a os consumidores tendem a crer que suas percepções de realidade são as únicas visualizações precisas, enquanto que quem discorda é visto como desinformado, irracional ou enviesado, e, o viés de confirmação em que consumidores preferem receber informações que confirmam suas visões de mundo. O artigo sugere usar informações como o engajamento social do usuário e das postagens para ajudar a tomar uma decisão, mesmo assim é uma tarefa desafiadora pois o engajamento social dos usuários com as fake news gera dados volumosos, incompletos, não estruturados e ruidosos.

O artigo [11] propõe uma arquitetura de Rede Convolucional otimizada para detecção de fake news. É um modelo de aprendizagem supervisionada, e é chamado de OPCNN-Fake, em inglês *an Optimal CNN model for Fake news detection*. Este modelo usa várias camadas para extrair características de alto nível e de baixo nível. A otimização do modelo se fez usando a técnica de otimização *hyperopt* para selecionar os melhores valores para os hiperparâmetros. Foram usados para treino e teste quatro conjuntos de dados padrão: FakeNewsNet, FA-KES5, ISOT e um conjunto de dados do Kaggle que foi chamado apenas de Dataset1; de cada conjunto de dados foram retiradas amostras que correspondem a fração de 80% para treinamento e 20% para teste, e também foi utilizado validação cruzada 10-fold no treinamento. O modelo foi avaliado usando métricas padrões: acurácia, precisão, revocação e *F1-score*. Essas métricas foram usadas para compará-lo contra quatro outros modelos de aprendizagem profunda (LSTM de uma camada, LSTM de duas camadas, RNN de uma camada, RNN de duas camadas) e Regressão logística, K vizinhos mais próximos (K Nearest Neighbor – KNN), Floresta Aleatória, Máquina de Vetores de Suporte e *Naive Bayes*. A arquitetura OPCNN-Fake teve resultados melhores contra todos esses modelos em todos os quatro *datasets* utilizados, tanto no conjunto de treinamento com validação cruzada, quanto no conjunto de teste. Por esse motivo a arquitetura de classificação utilizada neste trabalho de conclusão de curso foi baseada no OPCNN-Fake.

A cada dia que passa surgem modelos de aprendizado de máquina cada vez mais complexos, urgindo a necessidade de grandes conjuntos de dados rotulados. No artigo [10] é proposto um framework baseado em supervisão fraca, que podem rotular grandes datasets rapidamente, fornecendo rótulos mais ruidosos porém mais baratos que o rotulamento manual. Essas fontes de supervisão fraca têm acurácias diversas e desconhecidas, podem fornecer rótulos correlacionados e podem rotular diferentes tarefas ou aplicar em diferentes níveis de granularidade. O framework promete integrar e modelar essas fontes de supervisão fraca, considerando-as como rótulos de diferentes sub-tarefas relacionadas de um problema. O modelo de rotulagem utilizado nesta pesquisa, LabelModel[14], é baseado nesta abordagem. Ele aprende um modelo estatístico que estima as probabilidades de rótulo a partir das informações das fontes ruidosas de rótulos.

3 METODOLOGIA

Nesta seção encontra-se a descrição do conjunto de dados usados nessa pesquisa, que inclui *tweets* e notícias de checagem de *fake news*, o processo de limpeza e exploração dos dados, o agrupamento das notícias para selecionar palavras-chaves importantes do contexto, do rotulagem da base de *tweets* usando um modelo de aprendizado com supervisão fraca, e por fim, a descrição do modelo de classificação da presença de *fake news* num *tweet*.

3.1 Descrição da base de dados de *tweets*

A base de dados de *tweets* [5] utilizada nesta pesquisa nos foi disponibilizada pela equipe INTERFACES do Departamento de Ciências Sociais da Universidade Federal de São Carlos que fez todo o processo de coleta dos *tweets* a partir do dia 20 de Junho de 2022 até o dia 24 de Fevereiro de 2023. Cada tweet, além do texto, contém um identificador único, identificador do autor, data de criação, fonte (mobile, web, *etc.*), linguagem, contagem de *likes, retweets* e de *quote retweets*, lista de usuários mencionados, lista de hashtags, lista de urls, (as 3 listas podem ser vazias, que é o mais comum) e por fim, identificador de conversa e/ou identificador de referência; cada *tweet* tem seu próprio identificador de conversa, que serve para ser referenciado (através do identificador de referência) por um tweet-resposta, e o tweet-resposta por sua vez pode ser respondido

[10] https://projetocomprova.com.br/
[11] https://lupa.uol.com.br/
[12] https://g1.globo.com/fato-ou-fake/
[13] https://sites.google.com/view/detector-de-fake-news/p%C3%A1gina-inicial?pli=1
[14] https://snorkel.readthedocs.io/

do mesmo modo. Bem como há uma base com usuários únicos que são autores dos *tweets* da primeira base, nela contém o identificador único de usuário (que é referenciado na base de *tweets* como id de autor), nome de usuário, data de criação da conta, se a conta tem proteção, localização informada na conta, contagem de seguidores, contagem de contas que o usuário segue, contagem de postagens, e por fim se a conta é verificada ou não – durante o período de coleta não existia "verificado pago", que é a assinatura do Twitter Blue[15], uma mensalidade que o usuário pode pagar para ter alguns benefícios na plataforma, até 2022 exostia apenas o símbolo de "verificado pela plataforma".

O corpus conta com mais de 200 milhões de tweets não rotulados, majoritariamente em português do Brasil, todos os *tweets* estão relacionados às eleições presidenciais de alguma forma, porém em contextos diferentes. Esses tweets foram coletados através da API do Twitter usando como filtro palavras chaves relacionadas a tópicos relevantes das eleições:

- Os presidenciáveis mais relevantes: Bolsonaro, Lula, Ciro Gomes e Simone Tebet – Período 20/06/22 a 31/01/23
  - Visando ampliar a abrangência na coleta de *tweets*, as consultas foram feitas com os nomes dos candidatos, apelidos populares e a hashtag #nome_do_candidato. Além disso, se o *tweet* for uma resposta ou um *retweet* com comentário, então também coleta-se o tweet que foi referenciado por ele (mesmo se não tiver nenhum apelido de candidato).

```
query_bolsonaro : #bolsonaro OR jair OR
    bolsonaro OR bozo OR biroliro OR "
    tchutchuca do centrao" OR bonoro OR
    capitao OR genocida OR mito OR bolsomito
     OR bolsolixo OR bolsotrump OR messias
    OR patriota OR b22 OR b17 OR brocha OR
    imbrochavel OR ma'conaro
query_lula : #lula OR lula OR "ex presidiario
    " OR lulalivre OR "9 dedos" OR luladrao
    OR lulaladrao OR lulinha OR nine OR luis
     inacio OR cachaceiro OR "sapo barbudo"
    OR lulao OR l13 OR "faz o L" OR lulindo
    OR metalurgico OR lulalkimin
query_ciro : #ciro OR ciro OR c12 OR
    cirogomes OR "ciro gomes" OR "correu pra
     paris" OR bolsolula OR ciranha
query_simone : #simone OR "simone tebet" OR
    simonetebet OR tebet OR "simone tablet"
    OR estepe OR s15
```

- Perfil dos candidatos: foram coletadas todas as postagens dos quatro presidenciáveis mais relevantes com referências** às respostas dessas postagens - Período 20/06/22 a 31/01/23
  - A API do Twitter disponibiliza um identificador único chamado de `conversation_id` que corresponde uma postagem às respostas/comentários (*quote retweet*) que ela receber. As respostas referenciam a postagem com a variável `reference_id`.
- Número dos candidatos do segundo turno: 13 e 22 – Período 01/10/22 a 31/10/22
- Pós eleição: devido aos comentários sobre alegação de fraude na contagem dos votos, um possível golpe de estado arquitetado pelo candidato perdedor ou pelos seus eleitores, sobre intervenção militar e afins – Período 30/10/22 a 24/02/23
- Atos golpistas: sobre a revolta e a invasão do Palácio do Planalto pelos eleitores de Bolsonaro em Janeiro após a posse do atual presidente Lula - Período 01/01/23 a 12/02/23

Optou-se por não usar todos os dados da base por alguns motivos. Primeiramente, para manter um escopo mais bem definido: pois há vários sub-tópicos, apesar de todos estarem ligados às eleições de 2022 de alguma maneira. Portanto, não serão abordadas as discussões sobre pós-eleições nem sobre os atos golpistas. Em segundo lugar, para reduzir a quantidade de dados a serem processados. Terceiro, não foi considerado o subgrupo do número dos candidatos. A princípio, parece uma boa ideia e que faz sentido com o proposto. A consulta é simplesmente "13" ou "22". Por sua natureza, é uma consulta extremamente abrangente, retornando bastante "lixo", *i.e.*, tweets sem relação alguma com as eleições. Portanto, precisaria de uma extensa limpeza dos dados, sobrando pouca coisa aproveitável. Além disso, o total bruto de dados era cerca de 4 milhões de tweets, que, em comparação com as outras consultas, é um número bruto muito pequeno. Logo, havia pouco custo-benefício em usar esse subconjunto. Então, dentre os sub-tópicos mencionados acima, o foco são as postagens que mencionam os protagonistas, Bolsonaro e Lula, que são as consultas `query_lula` e `query_bolsonaro`. A média de *tweets* diários do mês de Outubro é claramente maior que o restante do período observado, veja na Figura 1. O total bruto de *tweets* sobre Lula e sobre Bolsonaro somente do mês de outubro é de quase 27 milhões de tweets.

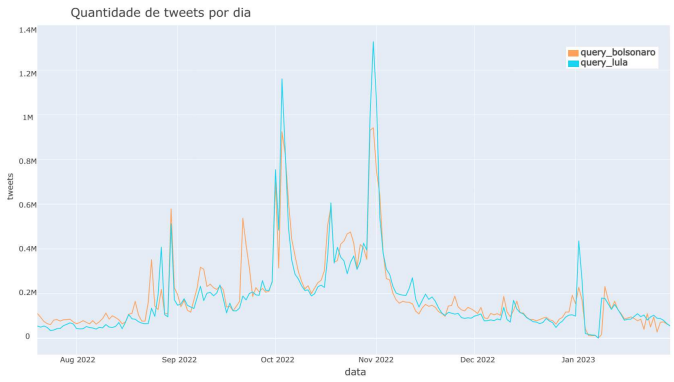

Figure 1: Quantidade de *tweets* coletados por dia de postagem. A linha laranja indica a quantidade de tweets citando o Bolsonaro, e a azul indica a quantidade de tweets citando o Lula.

A base utilizada contém exatamente 26.900.769 tweets antes da limpeza, são 13.358.707 tweets que mencionam o Lula, e 13.542.062 que mencionam o Bolsonaro. Já era esperado que grande parte desses tweets tivessem 0 likes, e de fato a mediana é 0 *likes, retweets* ou *retweets* com comentário, mais estatísticas na Tabela 1. Preferiu-se fazer uma filtragem por tweets com 10 ou mais likes e pelo menos 2 retweets, restando no total 2.100.629 tweets de 235.114 autores diferentes (em média 8,9 tweets por usuário), por dois motivos, o

[15] https://help.twitter.com/pt/using-twitter/twitter-blue

principal objetivo das fake news é o espalhamento, se não teve engajamento, não atingiu seu objetivo e não nos é interessante; segundo, para economizar recursos computacionais. As novas estatísticas após a filtragem encontram-se na Tabela 2.

Table 1: Estatísticas antes de filtrar os dados

Estatística	Tipo	Valor
Média	Like	1294
	Retweet	39
	RT c/ coment.	6
Desvio padrão	Like	5511
	Retweet	696
	RT c/ coment.	228
Percentil 25%	Like	0
	Retweet	0
	RT c/ coment.	0
Mediana	Like	0
	Retweet	0
	RT c/ coment.	0
Percentil 75%	Like	2
	Retweet	0
	RT c/ coment.	0

Table 2: Estatísticas após filtrar os dados

Estatística	Tipo	Valor
Média	Like	3365
	Retweet	517
	RT c/ coment.	65
Desvio padrão	Like	15090
	Retweet	2489
	RT c/ coment.	527
Percentil 25%	Like	37
	Retweet	5
	RT c/ coment.	0
Mediana	Like	162
	Retweet	18
	RT c/ coment.	2
Percentil 75%	Like	1017
	Retweet	128
	RT c/ coment.	11

Para esta pesquisa, é necessário dividir o conjunto de dados em várias etapas. Primeiramente, foi feita a separação do dataset para ser utilizado no classificador e no modelo de rotulagem com o auxílio da função `train_test_split()` do Scikit-Learn. Dividiu-se o conjunto de dados em 80% (1.680.503) para o classificador e 20% (420.126) para o rotulador. O conjunto do classificador será novamente dividido na proporção de 80/20, para ser usado nos estágios de treinamento e teste, respectivamente. Já o conjunto do rotulador foi dividido da seguinte forma: 1046 amostras foram rotuladas manualmente (de 1050, 4 amostras foram descartadas por ambiguidade no rótulo), e o restante dos dados destina-se ao treinamento do modelo. O modelo rotulador adotado é uma abordagem de supervisão fraca, baseada na biblioteca Snorkel [16] para Python3. Após ser treinado, o modelo rotulador é utilizado para rotular 100% do conjunto de dados do classificador, e algumas amostras não foram rotuladas (quando o modelo se abstem de atribuir um rótulo), isto é detalhado na **seção de rotulagem dos tweets**.

Quanto à rotulagem manual, foi necessário que a autora do presente trabalho realizasse a rotulagem manual de algumas amostras da base de dados para validar o modelo de rotulagem. Embora não seja a abordagem ideal para essa avaliação, foi necessário devido à limitação de tempo e à falta de recursos adicionais. Para determinar se um tweet continha uma "notícia falsa" ou não, foram considerados os seguintes aspectos:

- **A definição de uma *fake news***: afirmações falsas que, sobretudo, trazem algum tipo de vantagem política a algum dos candidatos.
- **Pesquisa**: algumas alegações de caráter duvidoso tiveram que ser pesquisadas e só foram descartadas ou confirmadas como *fake news* quando encontrada alguma reportagem em plataforma de notícias *renomadas*, *i.e.*, portais que são tidos como confiáveis pois têm histórico de publicar reportagens verdadeiras. Por exemplo: g1, globo news, uol confere, fato ou fake, agência lupa.

3.2 Descrição da base de notícias de checagem de *fake news*

Esta base, igualmente disponibilizada a nós pelo Departamento de Ciências Sociais da Universidade Federal de São Carlos, contém notícias em sua maioria do Uol Confere e algumas do Projeto Comprova, no total há 371 matérias únicas checando a veracidade de notícias de caráter duvidoso que se espalharam por redes sociais. Essas notícias foram de enorme importância para este trabalho, fornecendo valiosos *insights* para a criação de heurísticas usadas no modelo de rotulagem. A base contém: o título da checagem, data de publicação, veracidade da notícia, o candidato favorecido pela notícia falsa (se houver), agência de publicação, e uma url para a publicação. A partir disto, foi feita uma raspagem das notícias, ou *scraping*, com as bibliotecas de Python3: Requests e BeautifulSoup, em seguida foram removidas notícias duplicadas e foi feito um pré-processamento textual: remover tags HTML, remover pontuação, remover *stopwords* e fazer tokenização do texto (NLTK[17]).

Após o processamento dos textos, foram criados os vetores de *embeddings* com **Word2Vec**[7] usando a biblioteca de Python3 *Gensim*[18]. *Embeddings* são representações numéricas densas de palavras em um espaço vetorial contínuo de alta dimensão; durante o treinamento do modelo de *embedding*, o modelo analisa o contexto em que cada palavra aparece, palavras que são frequentemente usadas em contextos semelhantes tendem a ter representações de *embeddings* mais próximas umas das outras no espaço vetorial, dessa forma, o espaço vetorial gerado consegue representar numericamente informações semânticas e sintáticas dessas palavras. Os vetores foram criados com 100 dimensões. Verificando assimetria `dataframe.skew()` e curtose `dataframe.kurt()`, as distribuições

[16] https://github.com/snorkel-team/snorkel
[17] https://www.nltk.org/api/nltk.tokenize.html
[18] https://radimrehurek.com/gensim/models/word2vec.html

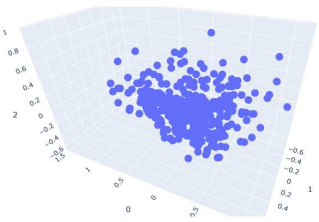

Figure 2: Visão frontal dos dados reduzidos com KPCA

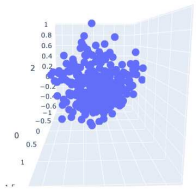

Figure 3: Visão lateral dos dados reduzidos com KPCA

dos vetores de *embeddings* aparentam ser similar à normal padrão dado que ambas as estatísticas foram próximas a 0 em todos os vetores – segundo a definição do Pandas.

Decidiu-se usar um algoritmo de clustering para agrupar as notícias e identificar automaticamente tópicos que poderiam ser utilizados na criação das funções de rotulagem. Seguindo a tabela do Scikit-Learn[19] na Figura 4, utilizou-se o Kernel PCA para reduzir a dimensionalidade dos vetores de embeddings das notícias para três dimensões, a fim de visualizar os dados em um gráfico, Figuras 2 e 3. Pelo gráfico, foi possível descartar o DBSCAN e o OPTICS, pois os dados se assemelham mais à terceira linha da tabela da Figura 4. O algoritmo BIRCH é mais adequado para datasets grandes, enquanto o MeanShift não lida bem com dados de alta dimensão, sendo assim, foram descartados. O AgglomerativeClustering, o WardClustering e o SpectralClustering realizaram, respectivamente, 3, 2 e 3 agrupamentos, como é característico desses algoritmos fazerem poucos clusters. No entanto, isso não estava alinhado com o objetivo de buscar vários tópicos distintos para a criação das funções de rotulagem. Essa necessidade ficou clara quando comparados os resultados desses algoritmos com o K-means e o AffinityPropagation, que formaram bem mais que 3 grupos e se tornaram as duas melhores opções. A formação de 3 ou 2 grupos resultava em clusters genéricos, sem representar de forma adequada os diferentes tópicos.

Em resumo, o K-means[20] funciona da seguinte forma: escolhe-se um valor para K, o algoritmo seleciona K centroides e atribui

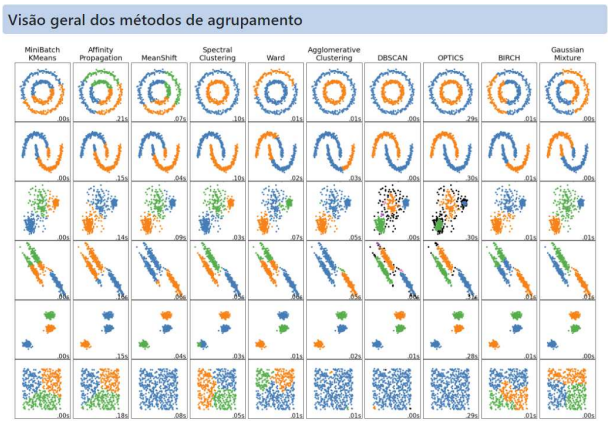

Figure 4: Comparação dos algoritmos de agrupamento do scikit-lean

os dados ao centroide mais próximo; os centroides são atualizados iterativamente até a convergência, movendo-se para a média dos pontos atribuídos a cada grupo. Enquanto o algoritmo Affinity Propagation[21] funciona da seguinte forma: é um método de agrupamento que identifica *automaticamente* os pontos mais representativos em um conjunto de dados para formar clusters. Ele utiliza medidas de afinidade entre os pontos para propagar a informação de similaridade e selecionar o exemplar mais representativo como centro do cluster.

Usando o método do cotovelo, aliado ao Silhouette score para cada k testado (de 2 a 30), o agrupamento ótimo foi obtido com $k = 11$ grupos, localizado no "cotovelo" do gráfico da Fig. 5 e com o maior silhouette score na Fig. 6 entre os valores de K próximos à dobra. Os agrupamentos do K-means apresentaram resultados satisfatórios, embora tenham ficado um pouco desbalanceados; isso, no entanto, não representou um problema para o meu objetivo final com os agrupamentos. O K-means criou um cluster de tamanho máximo com 60 notícias e um cluster de tamanho mínimo com 18 notícias. Por outro lado, os agrupamentos do Affinity Propagation mostraram-se mais promissores. Empiricamente, eles apresentaram uma distribuição mais uniforme e distinta. O Affinity Propagation gerou 29 clusters, dos quais 3 clusters possuíam o tamanho máximo de 31 notícias e 1 cluster possuía o tamanho mínimo de 1 notícia.

O algoritmo Affinity Propagation tem as seguintes características:

(1) Não requer a especificação prévia do número de clusters desejados.
(2) Pode lidar bem com dados que possuem estruturas complexas ou desconhecidas.
(3) O Affinity Propagation cria muitos clusters, e é capaz de lidar com clusters de tamanhos variáveis.
(4) É computacionalmente eficiente em conjuntos de dados relativamente pequenos a moderados, pois exige recursos computacionais significativos.

Essas características atenderam muito bem ao que era esperado, pois, vendo o gráfico gerado usando Kernel PCA, nas Figuras 2 e 3,

[19] https://scikit-learn.org/stable/modules/clustering.html#overview-of-clustering-methods
[20] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[21] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html

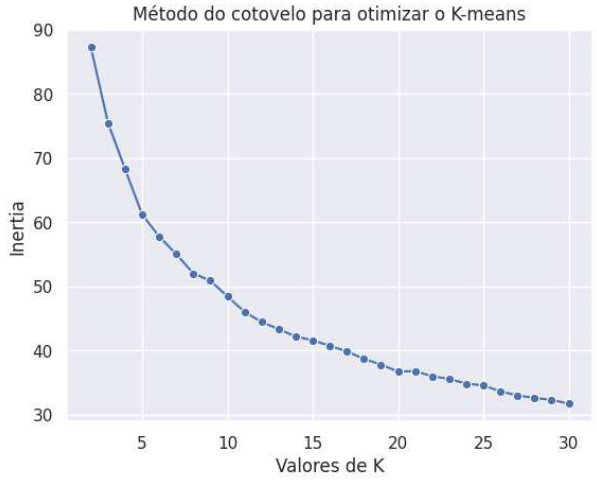

Figure 5: Método do cotovelo para o K-means

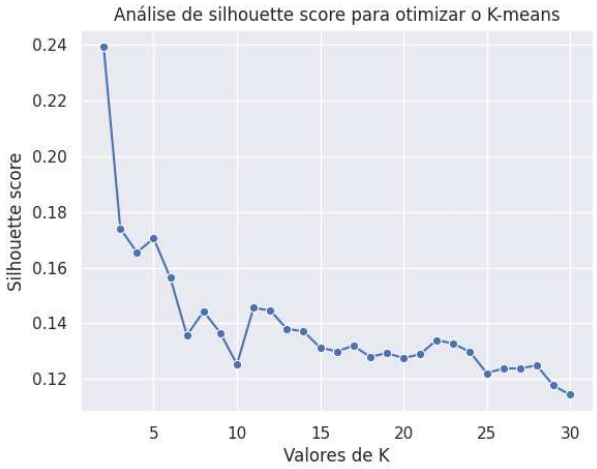

Figure 6: Silhouette score do K-means

não havia clusters bem definidos. Além disso, havia poucas notícias, o que tornou a complexidade do algoritmo irrelevante. Foi possível até mesmo utilizar o *Grid Search* para escolher o melhor valor de `damping`. A característica do algoritmo de identificar vários clusters foi exatamente o que o tornou superior aos outros métodos avaliados. Com os agrupamentos definidos, agora era necessário encontrar uma maneira de extrair os tópicos. Inicialmente, tentei obter os 10 termos mais representativos usando o Word2Vec e a função `word2vec.wv.most_similar`, utilizando o vetor que representava o centro de cada cluster. No entanto, os termos resultantes não foram satisfatórios em termos de fornecer ideias abrangentes. Uma alternativa simples, porém com resultados muito melhores, foi criar nuvens de palavras (Word Clouds) a partir de todas as notícias de um determinado agrupamento. Um exemplo pode ser observado na Fig. 7. Foram criadas nuvens de palavras para todos os agrupamentos que continham pelo menos dez notícias, o que totalizou dezessete gráficos, dentre vinte e nove, para análise visual.

Figure 7: Exemplo de Word Cloud feita com um dos clusters

Word Clouds, também conhecidas como nuvens de tags ou nuvens de palavras, são representações visuais em que o tamanho de cada palavra é proporcional à sua frequência de ocorrência em um texto específico ou em um conjunto de textos. Em outras palavras, palavras maiores representam maior frequência. Apesar da maior frequência, as palavras mais importantes na verdade são as palavras menores presentes nas nuvens. Isso ocorre porque as palavras mais frequentes em quase todos os textos tendem a ser as mesmas, como "vídeo", "presidente", "comprova", "lula", "bolsonaro" e "jair", veja a Fig. 7. Por outro lado, as palavras menores forneceram *insights* mais relevantes para a criação de funções heurísticas de rotulagem, como "vacina", "covid19", "fraude", "urna eletrônica", "militares", entre outras. É importante observar também que os quatro maiores agrupamentos apresentaram palavras mais genéricas e clusters pequenos demais, que foram os últimos a serem analisados já não traziam termos novos, por isso foram desconsiderados. Abaixo, as palavras extraídas dos word clouds, por cluster:

- CLUSTER 11: china, aborto, vídeo, motociata, controle natalidade, facada, pílula, coelho
- CLUSTER 15: laudo, empresa, Petrobras, terceirizada, processo eleitoral, serviço, urna, fraude eleitoral, maçonaria, carteiro/correio, direitos humanos
- CLUSTER 22: fábio assunção, ivermectina, militar, forças armadas, covid19, bndes, financiamento, mst
- CLUSTER 9: jogador, richarlison bolsonarista, vacina, tiktok e kwai, lava jato, collor
- CLUSTER 14: urnas, forças armadas, tiktok e kwai, indígena (terras indígenas), manifestações, luciferianismo, satanismo, igreja, professora
- CLUSTER 26: urna, indígena, ciro gomes, pesquisador(a/es), tiktok, polícia, londres
- CLUSTER 3: campanha, seções eleitorais, comício, boletins, urna, "lula xinga eleitores", ônibus, tiktok kwai facebook instagram youtube, alexandre moraes e xandão
- CLUSTER 6: bndes, jovem pan, gusttavo lima, ministério e ministro, heineken, jbs, moradia social, poliomielite, marine club, carne, frigorífico, seção eleitoral,urna, mato grosso, empresa, brasnorte, comício, vacinação
- CLUSTER 0: leite, carne, urna, boletins, conta, carros usados, aborto, china, pílula, portugal, pandemia, recontagem, ato, orçamento secreto, controle natalidade, imposto renda, pobre,

- CLUSTER 18: igreja, estádio, hospitais, pistola, torcida, vasco, enfermeiros, fraude, jovem pan, coronel tadeu, torcedores, youtube tiktok kwai, live
- CLUSTER 2: férias remuneradas, medida provisória mp, whatsapp tiktok kwai facebook, tv cultura, câmara, alexandre moraes (xandão), militares, benefício, forças armadas, jornalista, vera magalhães, jovem pan,
- CLUSTER 4: lulinha, filho, fazenda, ludmilla, argentina, google, caminhões, advogados, herança, condenado, tiktok kwai facebook, seção eleitoral, militar, manifestação
- CLUSTER 16: bahia, guerrilha, nikolas ferreira, militar, fidel castro, colômbia, rio de janeiro, salário mínimo, urna, complexo alemão, vila joão, fraude, medina, comando vermelho cv, sigla cpx cupinxa cupincha complexo, traficante
- CLUSTER 23: militares, eleitores, aposentadoria, benefício, forças armadas, roberto jefferson, andré, constituição, gleisi hoffmann, banheiro unissex, criança, xuxa, gazeta, segurança pública, previdência, facebook tiktok kwai, salário mínimo, grávida, bilionário, karina, gritos, mito, teto de gastos, elon musk
- CLUSTER 10: g1, imigrante, pará, calçada, namorar, guarapari, instituto, emprego, estrangeiro, igreja, neymar, religioso, google, drogas, MEI, pandemia, debate, teto de gastos
- CLUSTER 21: vacinação, mislav kolakusic, lacração, carga, terceirizado, sindicato, covid19, apoio, importações, parlamento europeu, pobre, morte, braga netto, janine small, seções eleitorais, criança, zona eleitoral, facebook tiktok kwai, 53 zona, vacina, itapeva, fraude, google
- CLUSTER 24: imunizante, janja, IPEC, ministro da educação, jean wyllys, canadá, mato grosso, pfizer, ianomâmi, miami, boca de urna, pesquisa boca, primeira universidade

3.3 Sobre a rotulagem dos *tweets*

O conjunto de tweets utilizado neste estudo não possui rótulos, o que apresenta um desafio para a aplicação de abordagens supervisionadas. No entanto, é importante ressaltar que a base de dados utilizada é extensa, contando com mais de 2 milhões de registros. Para contornar essa limitação, optaremos por uma abordagem de supervisão fraca, a fim de rotular esses dados antes de empregar um classificador supervisionado. A aprendizagem com supervisão fraca, também conhecida como "*weak supervision*" em inglês, é comumente utilizada quando é difícil ou custoso obter rótulos precisos para todo o conjunto de dados. Essa abordagem consiste em utilizar heurísticas, regras simples e potencialmente ruidosas para realizar a rotulagem dos dados. Nesse contexto, a biblioteca Snorkel[22], em Python3, tem como principal objetivo auxiliar desenvolvedores a criar um conjunto de treinamento que pode ser usado para treinar modelos de aprendizado. Embora seja mencionado que o Snorkel pode ser utilizado para criar um conjunto de treinamento sem a necessidade de qualquer rotulagem manual, no tutorial das funções de rotulagem, é recomendado ter um conjunto pequeno de dados rotulados manualmente para fins de avaliação final do modelo. Ressalta-se que esse conjunto não deve ser utilizado para análise de erros, mas sim para avaliação padrão de métricas, como acurácia e precisão, por exemplo.

Tendo como base os termos relevantes selecionados das notícias de checagem de *fake news*, foram criadas dezessete funções de rotulagem heurísticas para identificar as principais notícias falsas que marcaram o segundo turno das eleições:

- Tweets que relacionassem algum dos candidatos à maçonaria, satanismo ou canibalismo
- Alguma apologia ao "tratamento precoce" contra o Covid-19 e o "kit covid" de remédios sem comprovação científica, envolvendo palavras-chave como melhora, cura e afins
- Tweets afirmando fraude nas eleições ou que deem a entender que as urnas eletrônicas estejam beneficiando algum candidato com contagem de votos enviesada
- Tweets mencionando aborto e técnicas abortivas, nenhum dos dois candidatos mencionou apoiar o procedimento como método anti-concepcional durante o período de campanha eleitoral.
- Tentativas de relacionar Lula, seu filho, ou o Partido dos Trabalhadores a: traficantes, proposta de "kit gay" ou "forçar" ideologia de gênero nas escolas, aumento do desmatamento na Amazônia, fechar igrejas, entre outras notícias falsas conhecidas.
- TSE censurando alguém por motivos de disputa política, como se um candidato tivesse influência sobre o TSE para ganhar vantagem em campanha política
- O número de votar no candidato Bolsonaro é 17, o correto é o número 22.
- Algum jogador ou time de futebol fez campanha ou homenagem a algum candidato
- Financiamento ilegal de construções pelo BNDES

Outras heurísticas para rotular notícias falsas envolvem:

- Se o autor do tweet é conhecido por publicar desinformação com frequência, foram separados os seguintes usuários: @gazetadopovo, @revistaoeste e @PastorMalafaia
- Se contém palavras imperativas para o leitor compartilhar a informação, como por exemplo "espalhe", "compartilhe"

Enquanto as heurísticas para encontrar *tweets* sem notícias falsas:

- Tweet declarando voto
- Não mencionar o candidato por nome ou apelido amigável.
- Não mencionar o candidato por apelido pejorativo ou depreciativo
  - Apesar da busca na API ter sido especificamente por tweets que mencionem os candidatos, há vários *tweets* que não os mencionam de forma alguma, no caso de algumas opiniões falando somente sobre o contexto das eleições. Isso foi detectado durante a rotulagem dos tweets. Possivelmente são os comentários das publicações dos candidatos.
  - Essas heurísticas de não encontrar candidato no texto foram feitas com expressões regulares. Foram testadas duas implementações de reconhecimento de entidades, uma do Snorkel, e outra do Spacy[23], mas ambas ficaram ruins, possivelmente porque o texto é em português.
- Se o autor do tweet é um usuário dito confiável, seriam estes, portais de notícia ou jornalistas de renome.
  - Portais de notícia: @g1, @NexoJornal, @agencialupa, @exame, @aosfatos, @UOLNoticias, @GloboNews, @jornalnacional, @bbcbrasil, @portalR7, @CNNBrasil

[22] https://www.snorkel.org/get-started/

[23] https://spacy.io/models/pt

– Jornalistas: @miriamleitao, @evaristocosta, @cristilobo, @monicabergamo, @PVC, @PatriciaKogut, @LilianPacce, @gcamarotti, @AndreiaSadi, @JoycePascowitc

As funções de rotulagem são declaradas usando a anotação `@labeling_function()`, deve receber como entrada um texto x, é feita a verificação conforme a heurística, e retorna um valor inteiro: 1 para fake news, 0 não há fake news, -1 no caso do modelo não saber o que rotular. Por exemplo:

```
FAKE_NEWS = 1
NOT_FAKE = 0
ABSTAIN = -1

@labeling_function()
def check_canibalismo(x):
    if re.search(r"(canibal|carne humana)",
                 x.text.lower(), flags=re.I):
        return FAKE_NEWS
    else:
        return ABSTAIN
```

Usando um objeto do tipo *LFApplier* do Snorkel, aplicamos as funções, para cada entrada de dados há uma saída de cada função, retorna uma matriz relacionando todas as funções com todos os dados. Essa matriz deve ser dada como entrada a um modelo de rotulagem, o modelo principal do Snorkel é o LabelModel [10], que foi utilizado neste trabalho. Resumidamente, o LabelModel usa um *framework* probabilístico para combinar os rótulos gerados pelas funções heurísticas definidas, e assim gerar rótulos probabilísticos para os dados não rotulados.

Na Tabela 3 vemos a distribuição dos rótulos atribuídos pelo snorkel aos dados do modelo classificador. E na Tabela 4 vemos a distribuição dos rótulos atribuídos pelo snorkel aos dados durante o treinamento do modelo rotulador e também a distribuição dos rótulos que foram atribuídos manualmente para testar o modelo rotulador. Quatro *tweets* eram ambíguos em relação à sua veracidade, portanto foram **descartados** antes de fazer a avaliação do modelo de rotulagem. Pela maneira como foram definidas as funções de rotulagem, se o modelo não identificar se há *fake news* ou não, ele poderia abster-se de dar um rótulo, percebe-se que a proporção de abstenções do modelo nos dados não vistos (o conjunto do classificador) é a mesma que a proporção do conjunto de teste, por volta de 5%. Percebe-se também que os dados sofrem de sub-amostragem severa de notícias falsas, elas aparentam ser uma anomalia neste conjunto de dados. Dos dados que foram rotulados pelo modelo do Snorkel, todos tem entre 1% e 2% de notícias falsas em relação ao total. Mesmo no conjunto rotulado manualmente, na Tabela 4, a proporção de notícias falsas é bem pequena.

3.4 Sobre o modelo de classificação

O modelo de classificação utilizado teve sua arquitetura inspirada na arquitetura OPCNN-Fake proposta em [11], foi feita uma rede neural convolucional de treze camadas para processamento dos tweets implementada com a API de Deep Learnin para Python3 Keras[3], baseada no TensorFlow. Deve receber como entrada matrizes de números com tamanho fixo, por isso os *tweets* passaram por um pré-processamento antes de treinar o classificador. Utilizando o

Table 3: Distribuicão dos dados de treinamento e de teste do classificador

	Treinamento		Teste	
Rótulo	Freq.	Porc.	Freq.	Porc.
Fake	64488	5,134%	16414	5,230%
Não Fake	1256208	93,440%	313822	93,371%
Abster	23706	1,763%	5865	1,745%
Total	1344402	100%	336101	100%

Table 4: Distribuicão dos dados de treinamento e de teste (rotulamento manual) do modelo rotulador

	Treinamento		Rótulo Manual	
Rótulo	Freq.	Porc.	Freq.	Porc.
Fake	20394	5,212%	69	7,062%
Não Fake	391273	93,366%	977	93,048%
Abster	7409	1,768%	4	0,381%
Total	419076	100%	1050	100%

Tokenizer do Keras[24], cada *tweet* foi transformado numa matriz de números, e para fixar o tamanho das matrizes, encontra-se a maior matriz, e faz-se o padding com zeros nas outras matrizes para que todas tenham as mesmas dimensões $m \times m$. O modelo tem a seguinte estrutura: uma camada de embeddings, uma camada de Reshape, 3 sequências seguidas de: camada convolucional 2D, seguida por camada de MaxPooling2D, seguida por camada de SpatialDropout 2D, e depois das três sequências, uma camada Flatten e a Camada Densa de saída.

(1) **Camada de Embeddings**: Como parâmetro de pesos (*weights*) foi usada uma matriz de *embeddings* criada com glove.6B.200d [9], que cria vetores de 200 dimensões para cada palavra. A dimensão de entrada é o tamanho do vocabulário do conjunto de dados, a dimensão de saída é 200 que é a dimensionalidade do Glove, e o tamanho da entrada é a quantidade de palavras do maior tweet.
(2) **Camada de Reshape**: Redimensiona a saída da camada de *embeddings*, é necessário para que a saída seja aceita como entrada pela camada convolucional.
(3) **Camada Convolucional**: Camada Conv2D do Keras, recebe como entrada uma matriz de "palavras", essa camada é o componente fundamental desta rede neural porque ela é usada para extrair características relevantes das entradas. Todas as camadas usadas nessa rede foram definidas com 128 filtros, e tamanho de *kernel* 2, como descrito no artigo [11]. A função de ativaçao utilizada foi a ReLU (*Rectified Linear Unit*), essa função retorna somente valores positivos ou 0 se a entrada for negativa: $f(x) = max(0, x)$. Essa função de ativação é amplamente utilizada em redes neurais porque ajuda a resolver o problema de desvanecimento do gradiente, que pode ocorrer em funções de ativação saturadas, como a função sigmoide, evitando assim a desaceleração do aprendizado em camadas mais profundas da rede.

[24] https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer

(4) **Camada de Max Pooling**: A camada de pooling é uma janela que percorre o mapa de características, por ser max pooling, ela pega o maior valor dentro da janela, essa camada é usada para reduzir o mapa de características. Segundo o artigo [11], os autores usaram Max Pooling porque escolher o maior valor captura as características mais significativas do mapa.
(5) **Camada de Dropout**: Usar uma camada de dropout é uma conhecida técnica de regularização, serve para reduzir a complexidade do modelo e também diminuir o *overfitting*, o super ajuste dos pesos da rede neural que impede a capacidade de generalização de um modelo. Uma camada de dropout tradicional desativa *neurônios* de forma independente a cada atualização, enquanto que o spatial dropout desativa *regiões* de neurônios, ambos agem de forma aleatória. O uso do spatial dropout pode ser especialmente eficaz em redes neurais convolucionais, onde a estrutura espacial dos dados é importante para a extração de características.
(6) **Camada Flatten**: Serve para transformar a matriz em um vetor de uma dimensão, para dar como entrada à camada de saída
(7) **Camada Densa**: Camada de saída com ativação sigmóide, recebe como entrada o vetor unidimensional da camada anterior Flatten para produzir a resposta final com seu único neurônio, se o tweet contém *Fake News* (1) ou não (0).

Table 5: Hiperparâmetros do Classificador

Hiperparâmetro	Valor
Otimizador	ADAM
Função de perda	entropia cruzada binária
Tamanho de batch	512
Fração de validação	20%
Épocas	43 épocas (terminou com *early stopping*)
Callbacks	EarlyStopping (paciência = 3) ModelCheckpoint

4 RESULTADOS

4.1 Configurações utilizadas

Com exceção da coleta de notícias do Uol Confere/Projeto Comprova, todos os experimentos foram conduzidos utilizando *notebooks* no ambiente do Google Colab[25], utilizando a linguagem de programação Python 3. A versão gratuita do Google Colab possui uma limitação em relação à duração contínua da execução na mesma célula, uma vez que isso é considerado inatividade e resulta no encerramento automático do ambiente. Como resultado, não foi possível realizar o treinamento da rede neural utilizando a GPU Padrão, devido à duração prolongada do processo, que excederia o limite de tempo considerado como inatividade. O plano de assinatura do *Google Colab Pro* oferece uma alocação de 100 unidades computacionais por mês, e caso esse limite seja atingido, é possível adquirir 100 unidades adicionais pelo mesmo valor da assinatura. O

[25]https://colab.research.google.com/

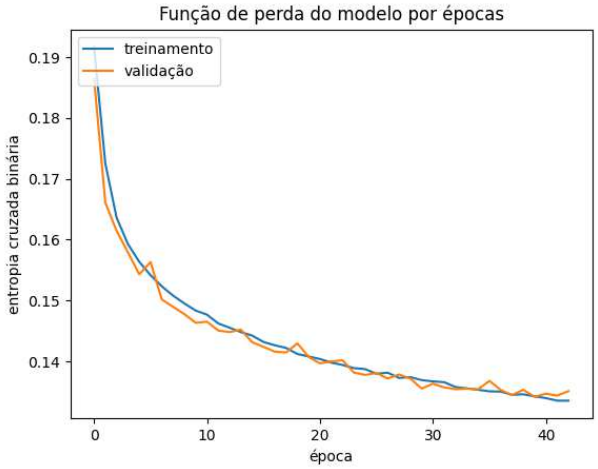

Figure 8: Função de perda (entropia cruzada binária) do modelo por épocas

treinamento do modelo de rotulagem teve uma duração aproximada de 3 horas, utilizando o ambiente com a GPU T4 e memória RAM de alta capacidade do Colab, enquanto o treinamento da rede neural de classificação levou cerca de 2 horas, utilizando o ambiente com a GPU A100 e memória RAM também de alta capacidade.

4.2 Observações e Métricas da Classificação

Nas Figuras 8 e 9 estão as métricas avaliadas durante o treinamento do modelo, a acurácia do conjunto de treinamento e de validação ficaram bem próximas, e a acurácia de validação oscila um pouco como visto na Fig 9. Pelo gráfico da função de perda na Figura 8, percebe-se que o gráfico ainda está em tendência de queda, apesar da perda no conjunto de validação aparentar uma subida, talvez voltasse a cair se a paciência do Callback de interrupção de treinamento estivesse maior que 3. Para não ultrapassar o limite de unidades computacionais da assinatura padrão, a melhor opção foi utilizar um valor baixo para a paciência do *callback* de interrupção

A acurácia do modelo no conjunto de teste foi de 96%. Na Tabela 7 vemos o valores das outras métricas de classificação da rede neural em relação aos dados de teste. Como mostrado na seção de rotulagem dos tweets, os dados sofrem de grave sub-amostragem, cerca de 5% dos dados (após eliminação das abstenções) têm rótulo positivo (*i.e., fake news*). O limiar de classificação escolhido foi **0,3**, foi levado em consideração que a rede neural aprendeu a classificar muito bem os rótulos negativos, mas não tão bem os rótulos positivos, com limiar = 0,5 a revocação de *tweets* com *fake news* ficou por volta de 0,25, e limiar = 0,2 baixava demais a precisão. Esse foi o limiar trouxe o melhor equilíbrio entre precisão e revocação, observando principalmente as médias macro e ponderada do *F1-score*.

A curva ROC – do inglês *Receiver Operating Characteristic* – é construída traçando a taxa de verdadeiros positivos e a taxa de falsos positivos. A área sob a curva ROC (AUC-ROC) é uma medida comumente usada para resumir o desempenho do classificador, quanto maior for o valor da AUC-ROC indica uma maior capacidade

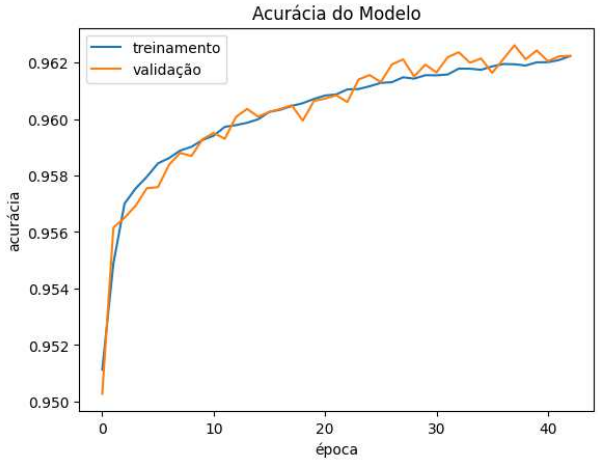

Figure 9: Acurácia do modelo por épocas

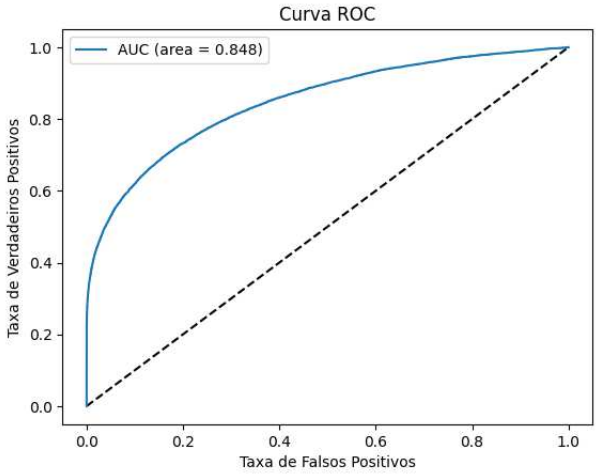

Figure 10: Área sob a curva ROC de 0.848

Table 6: Precisão, Revocação, *F1-score* nos dados de teste. Limiar de classificação = 0,3

Fake news?	Precisão	Revocação	*F1-score*	Qt. de Amostras
Não	0.97	0.99	0.98	313822
Sim	0.78	0.34	0.47	16414

Table 7: Médias macro e ponderada: Precisão, Revocação, e *F1-score* nos dados de teste. Limiar de classificação = 0,3

Médias	Precisão	Revocação	*F1-score*
macro	0.87	0.67	0.73
ponderada	0.96	0.96	0.96

de distinguir entre as classes positiva e negativa. A área sob a curva ROC foi de 0.848, é um valor bom, não é excelente, mas mostra potencial. Na Figura 10 mostra a área sob a curva ROC da rede neural de classificação, ela ficou acima da linha reta (AUC = 0.5) indicando que o modelo é melhor que um classificador aleatório.

4.3 Discussão sobre as classificações

Analisando uma amostra aleatória pequena dos dados (500 amostras) e comparando o rótulo atribuído pelo modelo rotulador, com a classificação predita e lendo o texto para avaliá-los manualmente, a grande maioria das postagens não contém notícias falsas, muitas vezes contém uma afirmação exagerada de uma opinião. Considerando a pequena amostra, a rede neural parece ter aprendido bem a identificar padrões subjacentes em uma notícia verdadeira, tiveram alguns casos de falsos negativos que aparentam ter sido mal rotulados pelo modelo de supervisão fraca, apesar de terem sido criadas poucas heurísticas para a rotulagem de postagens sem *fake news*. Nessa amostra não foi encontrado nenhum falso positivo em que a postagem aparentava ter notícia falsa, o classificador rotulou como falso e o rotulador não. O verdadeiro problema encontra-se nos dados em que o rotulador e o classificador concordaram que a postagem não continha fake news quando na realidade continha, foram encontrados alguns exemplos assim na amostra, da mesma forma encontrou-se alguns exemplos do rotulador e classificador concordando que uma postagem continha *fake news* que na realidade não contém, isso mascara a performance do modelo quando observamos as métricas como precisão e revocação, além disso, é provável que a proporção de falsos "verdadeiros positivos/negativos" seja similar no restante dos dados que não foram observados. Abaixo estão listados alguns exemplos:

Exemplo de verdadeiro positivo (rótulo: 1, predição: 1), mas que devia ser **não *fake news***, ambos erraram.

- " @taoquei1 MEI não é emprego, é um CNPJ entendeu? MEI não tem salário, não tem férias, 13°, repouso remunerado e nem FGTS. Ser MEI não é garantia de TRABALHO. Foi isso que Lula quis dizer e entendi, pois sou MEI. Te atualizei? "

Exemplo de Falso Positivo (rótulo: 0, predição: 1), em que devia ter sido considerado **não *fake news***, o rotulador acertou e o classificador errou.

- " PARA DEPUTADO FEDERAL É NENÉM DO TAPETE! PARA PRESIDENTE É LULA 13 Por São João de Meriti e pelo Brasil! https://t.co/0W4iKqVFaR "

Exemplo de Falso Negativo (rótulo: 1, predição: 0), que devia ser **não *fake news***, o classificador acertou e o rotulador errou.

- " Sou PcD e estou com Lula 13 https://t.co/dWdc4CvI9e "

Exemplo de Verdadeiro Negativo (rótulo: 0, predição: 0), mas que **é *fake news*** na verdade, ambos erraram.

- " Domingo Espetacular: "LULA é um dos MANDANTES do ASSASSINATO do prefeito Celso Daniel." Aponta investigação. Alexandre de Moraes já censurou a Liberdade da Record? @DomEspetacular @MPF_PGR @PoliciaFedeeral @STF_oficial @LulaOficial https://t.co/Fx1BBvRJAx "

5 CONCLUSÕES E TRABALHOS FUTUROS

Os autores reconhecem que não é adequado que uma única pessoa seja responsável pela rotulagem manual dos dados, porém, infelizmente, essa foi a alternativa disponível durante a pesquisa. Os resultados obtidos neste trabalho de conclusão de curso possuem como principal contribuição a metodologia que pode ser aplicada a outros contextos semelhantes. Pretende-se expandir o uso da metodologia proposta neste trabalho para futuras pesquisas, utilizando o mesmo conjunto de dados. No entanto, com mais tempo disponível, será possível ter amostras devidamente rotuladas por três pessoas diferentes e refazer o treinamento dos modelos de aprendizado.

O uso de um modelo de supervisão fraca para rotular grandes conjuntos de dados para posteriormente usá-los para treinar modelos de aprendizagem supervisionados não foi excelente mas pareceu bastante promissor. É possível desenvolver uma metodologia que seja objetiva para a criação de heurísticas e que seja baseada em dados, como foi o caso da criação de heurísticas com base em notícias de checagem de fatos.

Através de uma análise exploratória com algumas amostras de dados, o classificador parece ter lidado bem com a grande maioria dos exemplos que não contém *fake news*, tanto porque a maioria da base de dados é composta por esses exemplos, e porque a rede neural parece ter aprendido além de alguns rótulos ruins do rotulador. O rotulador mostra bastante potencial, e poderia ter sido melhor explorado, com a criação de mais funções heurísticas tanto para identificar notícias falsas quanto verdadeiras. Também através de funções de rotulagem mais elaboradas, como por exemplo, utilizando abordagens de análise textual e de sentimentos, isso foi pouco explorado devido a dificuldade em achar bons modelos pré-treinados em português.

6 AGRADECIMENTOS

Agradeço à minha família, meus pais Andréa e Cícero, meus irmãos Gabriela e Heitor, minha tia e meu padrinho Aldimere e Perasmo e meus primos Carolina e Gustavo, por todo o carinho e apoio durante minha jornada pela universidade desde que entrei na UFCG em Engenharia, até o meu último período em Computação. Agradeço ao corpo docente da UFCG pelo valioso aprendizado que absorvi ao longo desses anos, em especial ao meu orientador Eanes que me acompanhou ao longo de três PIBICs. Agradeço à Matheus Lisboa que tem me acompanhado desde o primeiro período, tem sido meu melhor amigo e meu grande amor desde então.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Matthew Allcott, Hunt e Gentzkow. 2017. Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives* 31, 2 (May 2017), 211–36. https://doi.org/10.1257/jep.31.2.211

[2] Hernán A. Bovet, Alexandre e Makse. 2019. "Influence of fake news in Twitter during the 2016 US presidential election". *"Nature Communications"* 10, 1 (2019), 7. https://www.nature.com/articles/s41467-018-07761-2

[3] François Chollet et al. 2015. Keras. https://keras.io.

[4] Célio Santana Júnior e João Pedro Albuquerque e Fabiola Queiroz e Steffane Lima. 2014. A disseminação da informação no Twitter: uma análise exploratória do fluxo informacional de retweets. *AtoZ: novas práticas em informação e conhecimento* 3, 1 (2014), 50–59. https://revistas.ufpr.br/atoz/article/view/41334

[5] INTERFACES. 2022. Dataset de tweets das eleições presidenciais brasileiras de 2022.

[6] Lima Menezes. 2022. A Pessoa Idosa no Combate às Notícias Falsas (Fake News): Uma Questão que Envolve a Todos Nós. *Revista Mais 60* 33, 83 (2022).

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs.CL]

[8] Márcia Nunes. 2010. *Rádio, cidadania, e campanhas eleitorais : (1998-2008).* e-papers.

[9] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[10] Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. 2018. Training Complex Models with Multi-Task Weak Supervision. arXiv:1810.02840 [stat.ML]

[11] Abdullah e Alsamhi Saeed Hamood Saleh, Hager e Alharbi. 2021. OPCNN-FAKE: Optimized Convolutional Neural Network for Fake News Detection. *IEEE Access* 9 (2021), 129471–129489. https://doi.org/10.1109/ACCESS.2021.3112806

[12] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake News Detection on Social Media: A Data Mining Perspective. arXiv:1708.01967 [cs.SI]