DISSERTAÇÃO DE MESTRADO PROFISSIONAL

STALLA: UM FRAMEWORK PARA ANÁLISE DE FONTES ABERTAS DURANTE A PANDEMIA DO COVID-19

ALEXANDRE CABRAL GODINHO

Programa de Pós-Graduação Profissional em Engenharia Elétrica
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

STALLA: UM FRAMEWORK PARA ANÁLISE DE FONTES ABERTAS DURANTE A PANDEMIA DO COVID-19

STALLA: A FRAMEWORK FOR OPEN SOURCE ANALYSIS DURING THE COVID-19 PANDEMIC

ALEXANDRE CABRAL GODINHO

Orientador: GERALDO PEREIRA ROCHA FILHO, Ph.D/UESB

Coorientador: VINÍCIUS PEREIRA GONÇALVES, Ph.D/UnB

PUBLICAÇÃO: PPEE.MP.056

BRASÍLIA-DF, JUNHO 2023

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO PROFISSIONAL

STALLA: UM FRAMEWORK PARA ANÁLISE DE FONTES ABERTAS DURANTE A PANDEMIA DO COVID-19

ALEXANDRE CABRAL GODINHO

Dissertação de Mestrado Profissional submetida ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Mestre em Engenharia Elétrica

Banca Examinadora

Prof. Dr. Geraldo Pereira Rocha Filho, UESB _____
Presidente

Prof. Dr. Fábio Lúcio Lopes de Mendonça, FT/UnB _____
Examinador Interno

Prof. Dr. José Rodrigues Torres Neto, UFPI _____
Examinador Externo

Prof. Dr. Edna Dias Canedo, CiC/UnB _____
Suplente

FICHA CATALOGRÁFICA

GODINHO, ALEXANDRE CABRAL
STALLA: UM FRAMEWORK PARA ANÁLISE DE FONTES ABERTAS DURANTE A PANDEMIA DO COVID-19 [Distrito Federal] 2023.
xvi, 51 p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2023).
Dissertação de Mestrado Profissional - Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. Redes Sociais	2. Redes Neurais Recorrentes
3. LSTM e BiLSTM	4. Supervisão Fraca
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

GODINHO, A.C. (2023). *STALLA: UM FRAMEWORK PARA ANÁLISE DE FONTES ABERTAS DURANTE A PANDEMIA DO COVID-19*. Dissertação de Mestrado Profissional, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 51 p.

CESSÃO DE DIREITOS

AUTOR: ALEXANDRE CABRAL GODINHO
TÍTULO: STALLA: UM FRAMEWORK PARA ANÁLISE DE FONTES ABERTAS DURANTE A PANDEMIA DO COVID-19.
GRAU: Mestre em Engenharia Elétrica ANO: 2023

ALEXANDRE CABRAL GODINHO
Depto. de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

DEDICATÓRIA

Inicialmente, dedico este trabalho a Deus, por me conceder saúde e resiliência nos momentos mais árduos, durante a realização do mestrado profissional.

Da mesma forma, dedico este trabalho à minha família, que sempre esteve ao meu lado com amor e apoio incondicionais. Obrigado por acreditarem em mim e por serem minha fonte de força e inspiração em todos os momentos.

Nesse sentido, que essa conquista sirva, também, de motivação para o meu filho Mateus, sempre que ele buscar alcançar os seus objetivos, na certeza de que o esforço e a dedicação investidos para o desenvolvimento de novos conhecimentos, o levará a produzir algo de positivo para a sociedade.

Por fim, dedico, ainda, este trabalho ao meu amigo Cristiano Nunes, pelo incentivo e pela contribuição com a pesquisa para a identificação de relevância de publicações em redes sociais, os quais foram fundamentais para que eu atingisse os objetivos propostos.

AGRADECIMENTOS

Gostaria de aproveitar esta oportunidade para expressar minha sincera gratidão a todas as pessoas que tornaram possível a realização deste trabalho e contribuíram para o meu crescimento acadêmico e pessoal ao longo desta jornada.

Primeiramente, desejo agradecer ao meu orientador, Prof. Dr. Geraldo Pereira Rocha Filho, e ao meu coorientador, Prof. Dr. Vinícius Pereira Gonçalves, por suas orientações sábias e incansáveis ao longo de todo o processo de pesquisa. Suas orientações perspicazes, valiosas sugestões e apoio contínuo foram fundamentais para o sucesso deste trabalho.

Agradeço também aos professores e membros da banca examinadora, Prof. Dr. Fábio Lúcio Lopes de Mendonça, Prof. Dr. José Rodrigues Torres Neto e Profª. Dra. Edna Dias Canedo, por dedicarem seu tempo e conhecimento para avaliar este trabalho e por suas contribuições construtivas que enriqueceram o conteúdo desta dissertação.

Minha gratidão se estende aos meus colegas do Programa de Pós-Graduação Profissional de Engenharia Elétrica (PPEE) da Universidade de Brasília (UNB), cujas discussões e trocas de ideias foram enriquecedoras e estimulantes. Agradeço pela amizade, pelo apoio mútuo e pela camaradagem que tornaram o ambiente acadêmico mais agradável e produtivo.

À minha família, que sempre acreditou em mim e me incentivou em todos os momentos, sou imensamente grato. Seu amor e encorajamento incondicionais foram a força motriz por trás das minhas conquistas.

Também desejo agradecer à Agência Brasileira de Inteligência (ABIN), por fomentar a pesquisa acadêmica em consonância com a Política e a Estratégia Nacional de Inteligência, no âmbito do Sistema Brasileiro de Inteligência (SISBIN).

Da mesma forma, quero agradecer à Chefia do Centro de Inteligência do Exército (CIE) pela disponibilização de infraestrutura de pesquisa e acesso a recursos indispensáveis para a realização deste trabalho.

Por fim, dedico uma menção especial aos amigos e pessoas queridas que, mesmo à distância, estiveram presentes, oferecendo palavras de incentivo e compreensão durante os momentos mais desafiadores.

Este trabalho não seria possível sem a colaboração e dedicação de cada um de vocês. Obrigado por fazerem parte desta jornada e contribuírem para o meu crescimento acadêmico e pessoal.

RESUMO

A expansão das redes sociais resultou em um aumento na distribuição de campanhas de desinformação, que colocam em risco a estabilidade democrática nacional, tornando-se um elemento desfavorável para a produção do conhecimento de Inteligência. Com o objetivo de mitigar este óbice, foi proposto o framework STALLA para coleta, tratamento, rotulação automatizada e análise de informações, proporcionando maior eficiência na produção do conhecimento. Assim, o estudo tem por escopo a pandemia do Covid-19, a partir de dados coletados de textos curtos (tweets), no idioma português, da rede social Twitter. Considerando-se os trabalhos correlatos, as Redes Neurais Recorrentes (RNN) apresentam-se como as mais vocacionadas para análises textuais. A partir dessa premissa, o desempenho do STALLA foi analisado comparando-se as implementações das redes LSTM e BiLSTM, resultando em uma acurácia de aproximadamente 70%, valor considerado expressivo para a definição da relevância da informação.

Palavras-chave: Redes Sociais, Redes Neurais Recorrentes (RNN), LSTM, BiLSTM, Supervisão Fraca.

ABSTRACT

The spread of social networks has resulted in an increase in the distribution of disinformation campaigns, which put national democratic stability at risk, becoming an unfavorable element for the intelligence knowledge production. In order to mitigate this bottleneck, the STALLA framework was proposed for the collection, treatment, automated labeling and analysis of information, providing greater efficiency in knowledge production. Thus, the study has as scope the Covid-19 pandemic, from data collected from short texts (tweets), in the Portuguese language, from the social network Twitter. Considering the related works, Recurrent Neural Networks (RNN) present themselves as the most suitable for textual analysis. Based on this premise, the performance of STALLA was analyzed by comparing the implementations of LSTM and BiLSTM networks, resulting in an accuracy of approximately 70%, a value considered significant for the definition of information relevance.

Keywords: Social Networks, Recurrent Neural Networks (RNN), LSTM, BiLSTM, Weak Supervision.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Siglas

API	*Application Programming Interface*
Bi-LSTM	*Bi-directional Long Short-Term Memory*
DL	*Deep Learning* (Aprendizagem Profunda)
DNN	*Deep Neural Network* (Rede Neural Profunda)
CNN	*Convolutional Neural Networks* (Redes Neurais Convolucionais)
CREDBANK	*Large-Scale Social Media Corpus with Associated Credibility Annotations*
FEVER	*Fact Extraction and Verification*
FN	*False Negative* (Falso Negativo)
FP	*False Positive* (Falso Positivo)
FPR	*False Positive Rate* (Taxa de Falso Positivo)
GPU	*Graphics Processing Unit* (Unidade de Processamento Gráfico)
GPT	*Generative Pre-trained Transformer*
IA	Inteligência Artificial
LSTM	*Long Short-Term Memory*
ML	*Machine Learning* (Aprendizado de Máquina)
MIT	*Massachusetts Institute of Technology*
MLOps	*Machine Learning Operations* (Implantação e Operação de Modelos de Aprendizado de Máquina)
NER	*Named Entity Recognition* (Reconhecimento de Entidade Nomeada)
OCR	*Optical Character Recognition* (Reconhecimento Óptico de Carácteres)
OSINT	*Open Source Intelligence* (Inteligência de Fontes Abertas)
PDN	Processo Decisório Nacional
PLN	Processamento de Linguagem Natural
RNA	Rede Neural Artificial
RNN	*Recurrent Neural Network* (Rede Neural Recorrente)
RMSprop	*Root Mean Square Propagation*
STALLA	*Scraping, Transforming, Auto-labelling, Learning and Analysis*
STT	*Speech to Text* (Transcrição de Áudio em Texto)
TN	*True Negative* (Verdadeiro Negativo)
TNR	*True Negative Rate* (Taxa de Verdadeiro Negativo)
TP	*True Positive* (Verdadeiro Positivo)
TPR	*True Positive Rate* (Taxa de Verdadeiro Positivo)

1 INTRODUÇÃO

A rápida expansão das redes sociais (1), associada à popularidade dos smartphones (2), permite que uma enorme quantidade de informação seja publicada diariamente. Nesse contexto, os usuários de redes sociais buscam informações nessas plataformas, deixando para um segundo plano a impressa. No entanto, a validação de veracidade dessas postagens é algo extremamente árduo, tendo em vista que é intencionalmente publicada para enganar as pessoas (3).

Assim, altera-se a percepção da sociedade sobre diversas temáticas, o que contribui para a divulgação de *fake news* (4) e campanhas de desinformação, provocando ameaças aos estados nacionais democráticos, bem como à segurança pública nesses países (5). A velocidade de propagação é o mais alarmante (6), sendo inegável que as redes sociais e as aplicações de mensagens instantâneas intensifiquem a divulgação de conteúdo enganoso (7).

Diante desse cenário, surgiram esforços para mitigar o crescimento das campanhas de desinformação (2). No entanto, em regra, as agências de *fact-checking* utilizam rotulação manual, realizada por analistas especializados, o que é um procedimento caro, demorado e pode ser contaminado pelo viés cognitivo do profissional que o realiza (1). Dessa forma, a comunidade de pesquisa identificou essa lacuna e vem desenvolvendo estudos com o objetivo principal de reduzir tempo e esforço humano.

Nesse contexto, pesquisas acadêmicas vêm contribuindo para: (i) identificar a velocidade de propagação das *fakes news* e campanhas de desinformação; (ii) analisar o comportamento dos usuários de redes sociais; (iii) propor frameworks capazes de verificar de forma automática a veracidade e a relevância dos conteúdos publicados em linguagem escrita; e (iv) explorar as características linguísticas que possibilitem identificar conteúdos enganosos (2).

Outrossim, o desenvolvimento de soluções tecnológicas, baseadas em quantidades massivas de dados e Inteligência Artificial (IA), enseja oportunidades para a atividade de Inteligência, como por exemplo: a automação da coleta de dados, a redução do tempo de processamento na análise de estruturas de dados complexas e o refinamento dos resultados para apresentar os principais pontos que conduzam a uma tomada de decisão efetiva por um gestor público (8, 9).

As principais técnicas de aprendizagem de máquinas para classificação de textos, incluindo modelos baseados em características e redes neurais, são fortemente orientadas por dados. Desta forma, os dados de treinamento constituem-se o primeiro requisito para a construção destes modelos, sendo que a qualidade desses dados se baseia em um conjunto equilibrado, suficientemente diversificado e cuidadosamente rotulado de artigos de notícias legítimas e falsas (10).

A fim de construir um sistema de classificação de texto para detectar conteúdos falsos, a partir de bases linguísticas, torna-se necessário que os artigos de notícias sejam avaliados individualmente e rotulados com relação ao seu nível de veracidade. Atualmente, dentre os conjuntos de dados elaborados e disponíveis, destacam-se: MisInfoText (10), LIAR (11), FEVER (12), Credibility Coalition Project (13), EMERGENT (14), PHEME (15) e CREDBANK (16). Esses últimos dois são específicos para os serviços de redes sociais, tendo sido criados para verificar a veracidade de tweets coletados.

É sabido que, com o passar dos anos, a sociedade contemporânea impõe novos desafios à produção de conhecimentos e ao assessoramento, no tocante ao Processo Decisório Nacional (PDN). Nesse contexto, o mundo contemporâneo amplia o papel da Inteligência de Estado, ao mesmo tempo em que impõe o desafio de reavaliação, de forma ininterrupta, no curso de uma crescente evolução tecnológica.

A atividade de Inteligência é caracterizada pelo exercício permanente e oportuno de ações especializadas, voltadas para a produção e difusão de conhecimentos, com o objetivo de assessorar o PDN, bem como identificar oportunidades e ameaças para o Estado Brasileiro (17).

O fenômeno conhecido como "information overload" afeta o processo decisório, pois provoca uma sobrecarga de informações para o analista e o decisor (18). Além do excesso de informações, há também a questão da experiência e do quadro de referência do analista de Inteligência, já que é uma tarefa com alto grau de subjetividade.

Os profissionais de Inteligência são treinados a desenvolver capacidades variadas, mas, muitas vezes, tornam-se vulneráveis a cometer erros de análise oriundos de vieses cognitivos (19, 20).

O desenvolvimento de soluções tecnológicas, baseadas em quantidades massivas de dados e IA, ensejam oportunidades para a atividade de Inteligência, como por exemplo: a automação da coleta de dados, a redução do tempo de processamento na análise de estruturas de dados complexas e o refinamento dos resultados para apresentar os principais pontos que conduzam a uma tomada de decisão efetiva (18, 21).

Nesse sentido, a automação da coleta de dados de mídias sociais, o tratamento e a análise de grandes volumes de dados (*Big Data e Analytics*) tendem a resultar em uma redução do tempo de processamento de estruturas de dados complexas.

1.1 JUSTIFICATIVA

A identificação de uma informação que possui potencial para mobilizar usuários da plataforma, aproxima-se da finalidade precípua da Atividade de Inteligência, que é obter o conhecimento em momento oportuno para subsidiar a tomada de decisão pelo gestor responsável. Dessa maneira, a pesquisa resultou no *Framework* STALLA que permite alcançar uma maior produtividade na análise de informações de fontes abertas por parte do analista de Inteligência.

1.2 OBJETIVOS

Diante do exposto, a pesquisa em questão, por intermédio da implantação de *weak supervision* e *Recurrent Neural Network* (RNN) com *Bidirectional Long Short-Term Memory* (BiLSTM), possui os seguintes objetivos precípuos: identificar as informações que geram maior engajamento e possuem maior potencial para fomentar campanhas de desinformação, bem como mitigar a carência de *datasets* temáticos, através da supervisão fraca. Para isso foi proposto o *Framework* STALLA para atenuar os efeitos da subjetividade e conferir maior assertividade para a produção de Inteligência.

1.3 CONTRIBUIÇÕES DO TRABALHO

Conclui-se que o modelo de predição treinado para detectar informações de maior relevância, no contexto da pandemia do Covid-19, atingiu aproximadamente 70% de acurácia, expondo a efetividade do *Framework* STALLA na identificação de informações, potencialmente, mais relevantes, mesmo não se tratando de informações necessariamente verdadeiras. A acurácia de 70% é exclusiva para o caso de estudo que se busca aplicar uma interpretação de valor automatizada com viés de interesse do analista para dados desconhecidos. Em outras temáticas e dependendo da qualidade das funções de classificação é esperado que o valor da acurácia varie. A temática da Covid é uma temática complexa com grandes divergências, o valor de 70% seria considerado baixo em outros contextos.

1.4 PUBLICAÇÕES RESULTANTES DESTA PESQUISA

Os principais resultados obtidos deste trabalho foram publicados nos Anais do VII Workshop de Computação Urbana (CoUrb), do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC) 2023, Qualis B2:

- GODINHO, Alexandre C. et al. STALLA: Um Framework para Análise de Fontes Abertas durante a Pandemia do Covid-19. In: Anais do VII Workshop de Computação Urbana. SBC, 2023. p. 54-67.
- GODINHO, Alexandre C et al. Análise preliminar da perspectiva cognitiva da dimensão informacional no conflito entre Rússia e Ucrânia através da aplicação de técnicas de aprendizagem de máquina de supervisão fraca. Brazilian Journal of Development, Curitiba, v. 9, n. 5, mai., 2023.

1.5 ESTRUTURA DA DISSERTAÇÃO

Ademais ao tópico introdutório, a dissertação está organizada da seguinte forma. O Capítulo 2 apresenta a fundamentação teórica relacionada à detecção de informações relevantes em fontes abertas a respeito da temática da pandemia do COVID-19, empregando técnicas de aprendizado de máquinas. O Capítulo 3 apresenta os trabalhos relacionados à detecção de informações relevantes em fontes abertas e suas principais limitações, as quais são objeto de pesquisa nessa dissertação. O Capítulo 4 apresenta a visão geral da estratégia proposta para a detecção de informações relevantes obtidas a partir de fontes abertas, tendo por base a *Open Source Intelligence* (OSINT), a rotulação de amostras por meio de supervisão fraca, e o STALLA. O Capítulo 5 apresenta a avaliação do STALLA, por intermédio de abordagens de aprendizado de máquinas, para que seja alcançada uma maior produtividade na análise de informações oriundas de fontes abertas, com o objetivo de subsidiar a tomada de decisão. Por fim, o Capítulo 6 apresenta as conclusões da pesquisa e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Para uma melhor compreensão da arquitetura proposta neste trabalho, voltada para a identificação das informações estatisticamente de maior relevância e que geram maior engajamento, bem como as que possuem maior potencial para fomentar campanhas de desinformação, faz-se necessário apresentar uma contextualização do arcabouço teórico que serviu de fundamento para o alcance dos objetivos. Nesse sentido, esta pesquisa fundamenta-se na aplicação de redes neurais artificiais à identificação de informações relevantes em postagens na rede social Twitter[1].

Nesse sentido, este capítulo apresenta uma introdução aos conceitos relacionados às redes neurais artificiais, à identificação de informações estatisticamente de maior relevância e à aplicação de redes neurais artificiais na identificação dessas informações. Por fim, são apresentadas as principais métricas para avaliação de desempenho dos modelos especificados para a identificação de informações relevantes.

2.1 REDES NEURAIS ARTIFICIAIS

Uma Rede Neural Artificial (RNA) é um sistema formado por unidades ou neurônios artificiais interligados em rede. De acordo com (22), um neurônio é uma unidade de processamento composta por ligações ponderadas (também chamadas de sinapses), um limiar de ativação (mais conhecido como *bias*), e uma função de ativação, normalmente não-linear. Um esquema básico é mostrado na Figura 2.1.

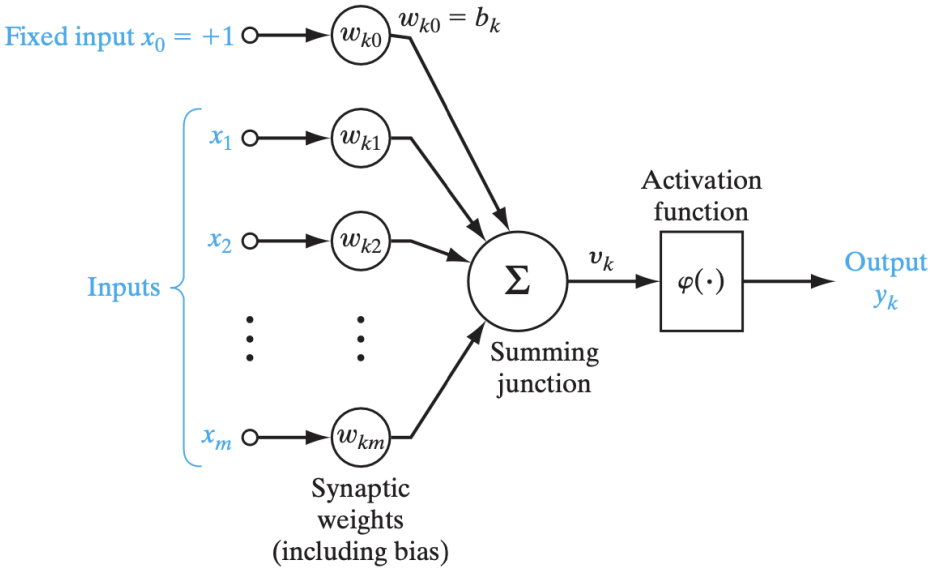

Figura 2.1: Modelo Conceitual de Neurônio Artificial.

O neurônio pode ser visto como um bloco que recebe o vetor de entrada $x \in \mathbb{R}\hat{}n$ e produz um valor intermediário de ativação a a partir de uma combinação linear da entrada com o vetor de pesos w e o limiar

[1] https://twitter.com

de ativação ou *bias* b. O valor a é passado como argumento para a função de ativação g, que produz a saída final y. Este comportamento é traduzido pelas equações 2.1 e 2.2, conforme (22).

$$a = \sum_{i=1}^{n} w_i x_i + b \tag{2.1}$$

$$y = g(a) \tag{2.2}$$

Em uma rede neural, cada neurônio pode ser explicado como uma regressão logística, com parâmetros $w = (w_1, w_2, ..., w_n)$ e b. Por intermédio de treinamento por retropropagação, (23) comprovou que a partir de cada neurônio, ou unidade, é capaz de extrair implicitamente uma determinada regularidade dos dados de entrada, possibilitando aos demais elementos da rede aumentar a expressividade em relação ao resultado do processamento.

Nesse contexto, conforme os diferentes neurônios se especializam durante o treinamento, as funções resultantes que a rede consegue representar recebem maior escopo. Esta medida de potencial representativo de uma rede é chamada de capacidade do modelo. Um esquema ilustrativo de uma rede neural é dado na Figura 9, onde cada círculo representa um neurônio como o da Figura 2.2 (22).

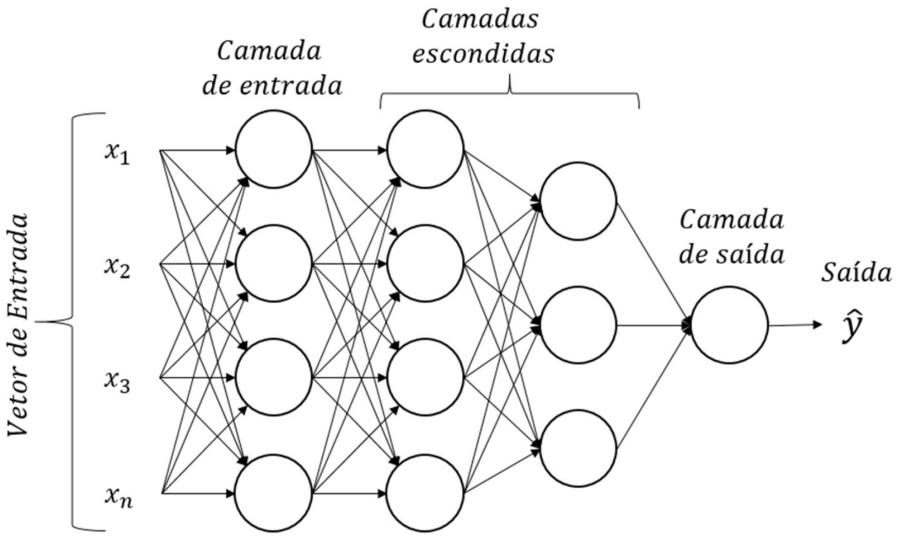

Figura 2.2: Rede Neural *Feed Forward*.

Via de regra, as redes neurais convencionais possuem uma estrutura composta por uma sequências de camadas. A primeira delas é a camada de entrada (*input layer*) e a última é a de saída (*output layer*). As camadas intermediárias são conhecidas como camadas ocultas (*hidden layers*). O número de neurônios em cada camada pode variar, dependendo do tipo de rede que está sendo construída.

A profundidade da rede (número total de camadas) e a cardinalidade de cada camada (número de neurônios de cada camada) são hiperparâmetros do modelo, representando os atributos que interferem no desempenho do modelo, porém que não são ajustados durante o treinamento. Uma rede neural é considerada uma rede rasa (*shallow neural network*) se tiver apenas uma camada intermediária, enquanto é

considerada profunda (*deep neural network*) se tiver duas ou mais camadas ocultas (22).

A saída final da rede, y, conforme a Figura 2.2, representa exatamente a predição gerada pelo modelo para a entrada x e os parâmetros w e b. É importante destacar que, em regra, y é um vetor e não um valor único, tendo em vista que a camada de saída da rede pode ter multiplos neurônios.

No caso de redes neurais sequenciais multicamadas, como a da Figura 2.2, o processo de cálculo da saída y é conhecido como *Forward Propagation* (Propagação Direta). Esta operação ocorre de forma sequencial entre as camadas, embora dentro de cada camada o processamento possa ocorrer em paralelo. Em outras palavras, as saídas de todos os neurônios de uma mesma camada podem ser avaliadas em paralelo, mas o processamento da camada anterior precisa estar concluído antes que a próxima camada da rede comece a ser avaliada (22).

Em (23), o erro de estimação é avaliado, após a geração da saída y, quando na oportunidade inicia-se o processo de ajuste dos parâmetros da rede neural. O procedimento estabelecido para esta função é o algoritmo de *back-propagation* (retropropagação). O nome da função decorre do fato de que o processo de cálculo é iniciado na última camada da rede, sendo propagado para as camadas anteriores.

2.2 APRENDIZAGEM PROFUNDA

De acordo com (24), Aprendizagem Profunda (*Deep Learning*) é a desginação para neurais com mais de uma camada intermediária. Assim, as redes utilizando as várias camadas de processamento não lineares são capazes de aprender em diferentes níveis de abstração.

Ainda segundo (24), as redes *Deep Learning* em decorrência de uma arquitetura mais robusta conseguem extrair melhor as características de um conjunto de dados. Atualmente, o emprego do aprendizado profundo é bastante popular devido aos seguintes motivos:

- o aumento vigoroso do poder computacional a partir da utilização de unidades de processamento gráfico (GPU);
- a expansão significativa do tamanho dos dados empregados para treinamento; e
- os recentes avanços em aprendizagem de máquina e processamento de sinais e informações que permitiram a construção de redes mais robustas e complexas.

2.2.1 Redes Neurais Recorrentes

Esta seção apresenta os conceitos a respeito de redes neurais recorrentes, tendo por referência o modelo básico, passando para camadas recorrentes mais recentes que são capazes de lidar com o aprendizado da memória interna, para lembrar ou esquecer certos padrões encontrados em conjuntos de dados. Nesse contexto da pesquisa, poderá ser verificado que as redes recorrentes são poderosas no caso de inferir padrões que são temporais ou sequenciais, e que permite uma melhoria no paradigma tradicional para um modelo que possui memória interna, sendo aplicável em ambas as direções no espaço temporal (25).

Dessa forma, com os resultados obtidos ao longo da pesquisa, será possível apresentar que um modelo bidirecional de memória de curto prazo longo pode representar uma vantagem sobre a abordagem direcional única, principalmente para aplicações para solucionar problemas de Processamento de Linguagem Natural (PLN), que é o caso para identificar as informações textuais de maior relevância.

As redes neurais recorrentes (RNNs) são baseadas nos primeiros trabalhos de Rumelhart (26). O conceito é simples, mas revolucionário na área de reconhecimento de padrões que utiliza sequências de dados.

O conceito de recorrência em RNNs pode ser ilustrado conforme mostrado na 2.3. Uma camada densa de unidades neurais podem ser estimuladas usando alguma entrada em diferentes intervalos de tempo, t. As Figuras 13.1 (b) e (c) mostram uma RNN com cinco intervalos de tempo, t = 5. Assim, é possível observar nas Figuras 13.1 (b) e (c) como a entrada é acessível para as diferentes etapas de tempo, mas mais importante, a saída das unidades neurais também está disponível para a próxima camada de neurônios (25).

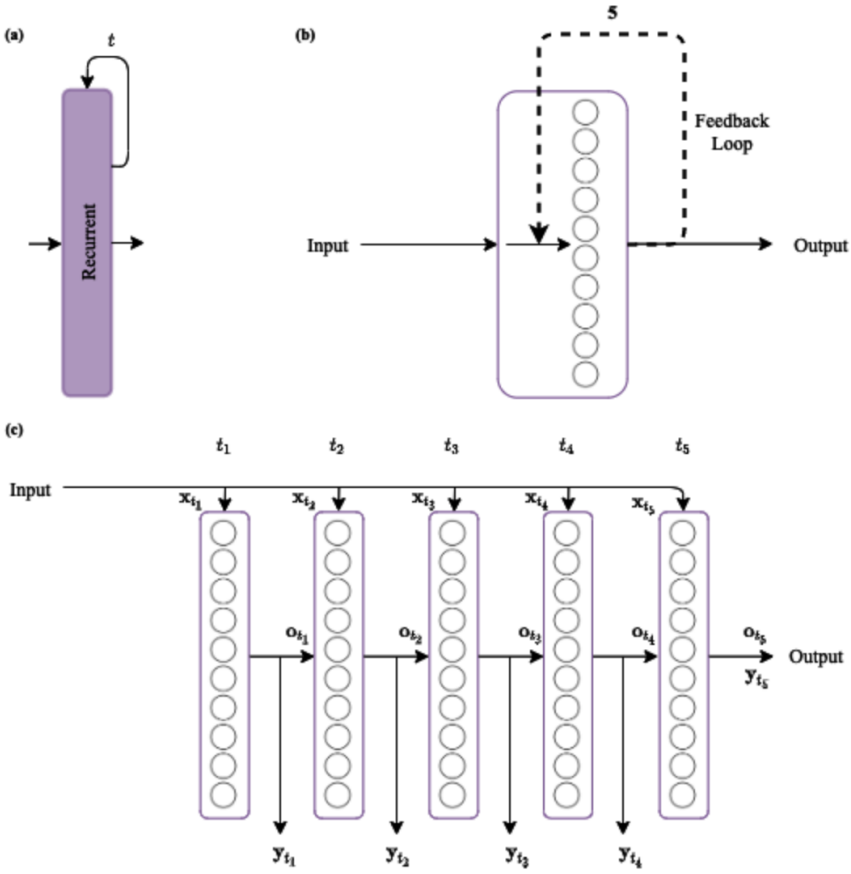

Figura 2.3: Diferentes representações de camadas recorrentes: (a) conceito de recorrência; (b) descreve as unidades neurais e o loop de feedback; e (c) é a versão expandida de (b), mostrando o que realmente acontece durante o treinamento.

A capacidade de uma RNN de ver como a camada anterior de neurônios é estimulada ajuda a rede a interpretar sequências muito melhor do que sem essa informação adicional. No entanto, isso tem um custo: haverá mais parâmetros a serem calculados em comparação com uma camada densa tradicional, devido ao fato de haver pesos associados à entrada x_t e à saída anterior o_{t-1} (25).

2.2.2 Long Short Term Memory (LSTM)

As redes *Long Short-Term Memory* (LSTM) são variações de redes neurais recorrentes (RNNs) que foram desenvolvidas para lidar com problemas que envolvem sequências de dados, como processamento de linguagem natural, reconhecimento de fala e tradução automática.

Inicialmente proposto por Hochreiter, os modelos de Memória de Longo Prazo Curto (LSTMs) ganharam força como uma versão melhorada de modelos recorrentes (27). As redes LSTM foram propostas para superar as limitações das RNNs tradicionais, que têm dificuldade em aprender dependências de longo prazo em sequências.

As LSTMs introduzem células de memória que permitem que as informações sejam armazenadas e acessadas por um longo período de tempo. Isso permite que as redes LSTM capturem dependências de longo prazo e evitem o problema do gradiente desvanecente, que ocorre quando os gradientes diminuem exponencialmente durante o treinamento de redes profundas (25).

As LSTMs são compostas por várias unidades de células de memória, cada uma com três portas principais: porta de entrada (*input gate*), porta de esquecimento (*forget gate*) e porta de saída (*output gate*). Essas portas controlam o fluxo de informações dentro da célula de memória e determinam quais informações serão armazenadas e esquecidas. A arquitetura das LSTMs permite que elas aprendam a reter informações relevantes e descartar informações irrelevantes nas sequências.

O diagrama na 2.4 mostra uma versão simplificada de um LSTM. Em (b), é possível observar o *self-loop* adicional que está ligado a alguma memória, e em (c), constatar como a rede se parece quando desdobrada ou expandida.

Há muito mais no modelo, mas os elementos mais essenciais são mostrados na 2.4. Nesse sentido, é possível observar como uma camada LSTM recebe do intervalo de tempo anterior não apenas a saída anterior, mas também algo chamado estado, que atua como um tipo de memória. No diagrama, pode-se visualizar que, embora a saída e o estado atuais estejam disponíveis para a próxima camada, eles também estão disponíveis para uso em qualquer ponto, se for necessário (25).

Há elementos que não estão exibidos na 2.4, dentre os quais se incluem os mecanismos pelos quais o LSTM recorda ou esquece de uma informação. Esses mecanismos são treináveis e otimizados para cada conjunto de dados de sequências, sendo os três principais componentes da rede os listados a seguir:

- **controle de saída**: quanto um neurônio de saída é estimulado pela saída anterior e pelo estado atual;
- **controle de memória**: quanto do estado anterior será esquecido no estado atual; e
- **controle de entrada**: quanto da saída anterior e do novo estado (memória) serão considerados para determinar o novo estado atual.

2.2.3 Bidirecional-Long Short Term Memory (BiLSTM)

Ao exarminar um LSTM bidirecional (BiLSTM), pode-se simplificar sua estrutura, considerando-se que é um LSTM que analisa uma sequência indo para frente e para trás, conforme mostrado na figura 2.5

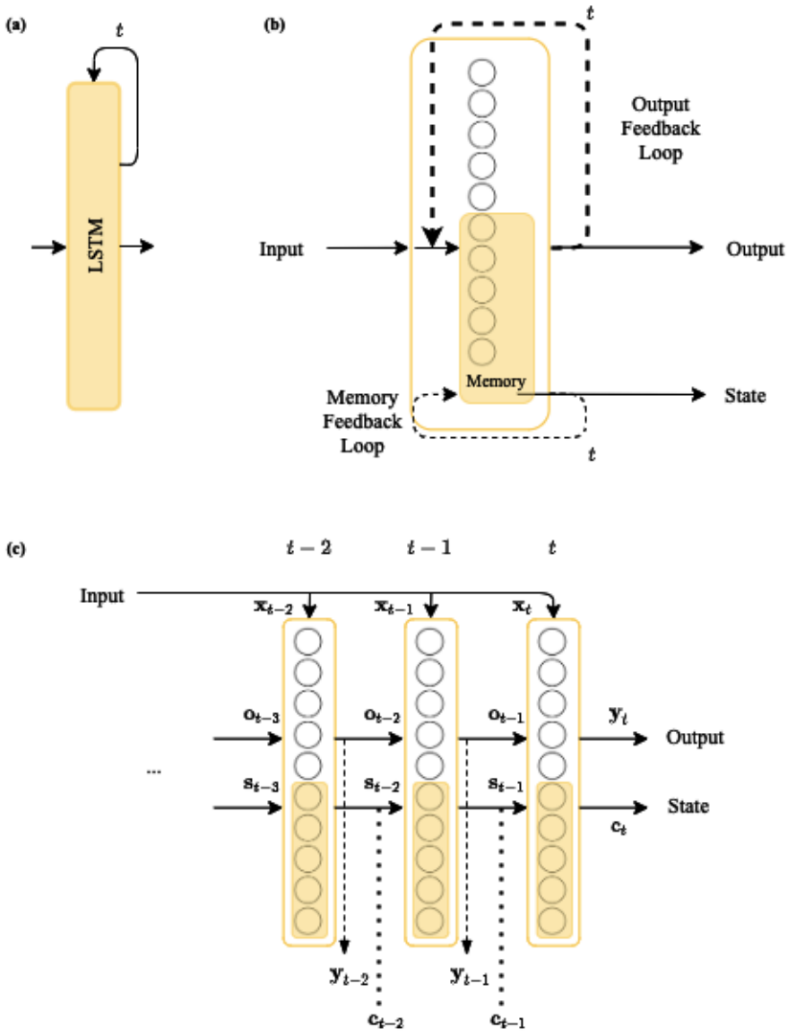

Figura 2.4: Representação Simplificada de uma Rede LSTM.

(25).

Na figura 2.5, também podemos observar que o estado e a saída de ambas as passagens, para frente e para trás, estão disponíveis em qualquer ponto da sequência. Para exemplificar a aplicação de um LSTM bidirecional, são apresentados os subsequentes empregos que uma análise de sequências percorrendo para frente e para trás poderá ter:

- Uma sequência de áudio que é analisada em som natural e depois retrocedendo, como forma de identificar mensagens subliminares.
- Uma sequência de texto, como uma frase, que é analisada quanto ao bom estilo percorrendo para frente, e também, posteriormente, no sentido contrário, já que alguns padrões linguistícos fazem referência para trás, principalmente no idioma português; por exemplo, um verbo que faz referência a um sujeito que aparece no início da frase.
- Uma imagem que tem formas peculiares indo de cima para baixo, ou de baixo para cima, ou de

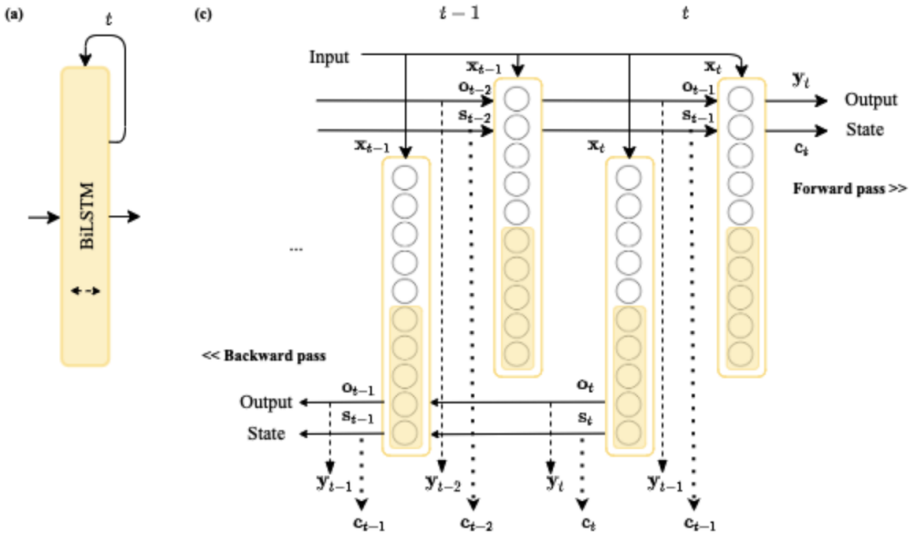

Figura 2.5: Representação LSTM bidirecional.

um lado para o outro e para trás; se você pensar no número 9, indo de cima para baixo, um LSTM tradicional pode esquecer a parte redonda no topo e lembrar a parte fina na parte inferior, mas um BiLSTM pode ser capaz de lembrar ambos os aspectos importantes do número indo de cima para baixo e de baixo para cima.

Por fim, pode-se resumir que as redes BiLSTM, por sua vez, são uma extensão das LSTMs que incorporam informações de contexto de ambos os lados da sequência. Enquanto as LSTMs processam a sequência apenas em uma direção (da esquerda para a direita ou da direita para a esquerda), as BiLSTMs usam duas camadas LSTM paralelas, uma processando a sequência da esquerda para a direita e a outra da direita para a esquerda. Isso permite que a rede capture tanto as dependências passadas quanto as futuras em relação a cada posição da sequência. As saídas das duas camadas LSTM são então combinadas para fornecer a saída final da BiLSTM.

As redes LSTM e BiLSTM têm sido amplamente utilizadas em várias aplicações de processamento de linguagem natural, como análise de sentimento, geração de texto, tradução automática e reconhecimento de entidades nomeadas. Sua capacidade de modelar dependências de longo prazo e incorporar informações de contexto de ambos os lados da sequência as torna poderosas para lidar com tarefas que envolvem dados sequenciais.

2.3 SUPERVISÃO FRACA

A supervisão fraca (*weak supervision*) é um termo usado em aprendizado de máquina para descrever situações em que os rótulos ou anotações fornecidos para treinar um modelo são menos precisos, incompletos ou menos específicos em comparação com a supervisão completa. Na supervisão fraca, os rótulos podem ser vagos, imprecisos ou apenas parcialmente disponíveis, o que pode ocorrer em várias situações, tais como:

- **Rótulos de classe parciais**: Em vez de ter rótulos precisos para todas as instâncias de treinamento, apenas algumas instâncias têm rótulos de classe disponíveis, enquanto outras podem ter rótulos ausentes ou desconhecidos.
- **Rótulos de classe imprecisos**: Os rótulos podem conter erros ou ambiguidades, o que significa que nem todos os rótulos estão corretos ou perfeitamente correspondentes às instâncias.
- **Rótulos de instância fracos**: Em vez de rótulos precisos, apenas informações parciais ou indiretas estão disponíveis para descrever as instâncias.
- **Rótulos de nível inferior**: Em vez de rótulos de alto nível ou categorias específicas, apenas rótulos de nível inferior ou informações mais genéricas estão disponíveis.

Assim, a supervisão fraca é um desafio para os algoritmos de aprendizado de máquina, pois o treinamento de modelos com rótulos fracos pode levar a resultados menos precisos ou menos confiáveis. No entanto, técnicas e abordagens específicas têm sido desenvolvidas para lidar com a supervisão fraca, como aprendizado semi-supervisionado, aprendizado ativo, aprendizado de transferência, aprendizado por reforço e métodos de inferência probabilística (28).

Essas abordagens buscam compensar a falta de supervisão completa, explorando outras informações disponíveis, incorporando conhecimento prévio ou adaptando os modelos para aprender com rótulos fracos de forma mais eficiente e eficaz. O objetivo é aproveitar ao máximo os dados disponíveis, mesmo que a supervisão seja menos precisa, para obter modelos úteis e generalizáveis (28).

A partir da apresentação dos modelos de aprendizado profundo, fica evidente que os mesmos fornecem resultados bastante precisos, sendo que suas arquiteturas eliminam a necessidade de engenharia de recursos, desde que com dados de treinamento suficientes. No entanto, para isso, é necessário que estejam disponíveis uma enorme quantidade de dados, para que esses modelos aprendam a estrutura subjacente dos dados (29). Dessa forma, a maior parte dos cientistas e dos engenheiros de dados, atualmente, depende de dados rotulados de qualidade para treinar modelos de aprendizado de máquina. Entretanto, construir um conjunto de treinamento para uma tarefa específica, de forma manual é demorado e caro, podendo ocorrer, ainda, o enviesamento do conjunto de treinamento em virtude do viés cognitivo do especialista que irá rotular o dado (28).

O custo de criação, limpeza e rotulagem de dados de treinamento geralmente é uma despesa significativa em termos de tempo e dinheiro. Além disso, em alguns casos, a privacidade é um requisito fundamental, e uma abordagem "sem olhar", em que os profissionais de aprendizado de máquina não podem inspecionar diretamente os dados para fornecer rótulos, não é possível devido a informações confidenciais e de identificação pessoal nos dados (28).

Diante do exposto, a supervisão fraca é uma ampla coleção de técnicas em aprendizado de máquina em que os modelos são treinados usando fontes de informações que são mais fáceis de se obter do que os dados rotulados manualmente, onde essas informações são incompletas, inexatas ou menos precisas.(28) Ao invés de um especialista em determinada temática rotular manualmente os dados de alta qualidade, mas com elevado custos, em termos de recursos humanos e financeiros, é pssível usar outras técnicas que combinam diversas fontes de dados, criando uma aproximação de rótulos. Dessa maneira, empregando-se

supervisão fraca, estrutura-se as rotulações em um único rótulo, permitindo que esses rótulos ruidosos e de origem fraca sejam combinados programaticamente para formar os dados de treinamento que podem ser usados para treinar um modelo (28).

Os rótulos são considerados "fracos" porque são ruidosos, ou seja, as medições de dados que os rótulos representam não são precisas e têm uma margem de erro. Assim, os rótulos também são considerados "fracos" se tiverem informações adicionais que não indiquem diretamente o que se quer prever (28).

O modelo criado usando dados de treinamento gerados por meio de supervisão fraca é comparável em desempenho a um modelo de aprendizado supervisionado criado com rótulos "fortes" tradicionais. Ademais, a partir das descobertas dos pesquisadores do *Massachusetts Institute of Technology* (MIT), o uso de uma combinação de alguns rótulos "fortes" associados a um conjunto maior de dados de rótulos "fraco" resultou em um modelo que não apenas aprendeu bem, mas também treinou em um ritmo mais rápido, obtendo-se um melhor desempenho (28, 30, 31).

2.4 FRAMEWORK SNORKEL

Nos últimos anos, houve uma explosão de interesse em sistemas baseados em aprendizado de máquina na indústria, nos governos e na academia, com um gasto estimado anual de US$ 12,5 bilhões. Nesse escopo, o desenvolvimento de técnicas de aprendizado profundo podem aprender representações específicas de tarefas de dados de entrada. Essas representações aprendidas são particularmente eficazes para tarefas como processamento de linguagem natural e análise de imagem, que têm entrada de alta variância e alta dimensão, sendo impossível de serem capturadas intregralmente com regras simples ou recursos projetados manualmente (30).

No entanto, o aprendizado profundo tem um elevado custo inicial, pois esses métodos precisam de conjuntos de treinamento massivos de exemplos rotulados para aprender, sendo geralmente dezenas de milhares a milhões para atingir o desempenho preditivo máximo. Assim, esses conjuntos de treinamento são extremamente caros para criar, especialmente quando é necessária experiência no domínio, pois são necessários especialistas no assunto (30).

Outra opção é a utilização de técnicas clássicas como *active*, *transfer* e *semi-supervised learning*. A maior parte dos profissionais está se voltando cada vez mais para alguma forma de supervisão fraca: fontes mais baratas de rótulos que são mais ruidosos ou heurísticos. No entanto, apesar de não apresentar custo elevado, muitas vezes têm precisão e cobertura limitadas (30).

A rotulagem de dados de treinamento é, cada vez mais, o maior óbice na implantação de sistemas de aprendizado de máquina. Com a intenção de mitigar esse obstáculo, a partir de uma pesquisa realizada na Universidade de Stanford, em 2016, desenvolveu-se o *Framework Snorkel* com a premissa de que os usuários poderiam rotular, criar e gerenciar dados de treinamento de forma programática. Em resumo, a proposta do *Snorkel* é permitir programaticamente que os usuários possam treinar modelos de última geração sem rotular manualmente quaisquer dados de treinamento (30, 31).

O *Snorkel* é um projeto de código aberto para rotulagem programática por meio de supervisão fraca,

que auxilia a enfrentar os desafios de Implantação e Operação de Modelos de *Machine Learning* (MLOps), como governança de modelo, operações, linhagem de dados, etc. Assim, a supervisão fraca expressa como código permite flexibilidade à medida que incorpora-se diferentes pontos de dados. Isso contribui para melhorar a generalização e pode ser dimensionado facilmente com dados não rotulados (28, 30, 31).

A partir do *Snorkel* os usuários podem criar modelos de aprendizado de máquina utilizando rótulos fracos, como heurísticas ou regras programáticas, em vez de depender exclusivamente de rótulos precisos e anotados manualmente por um especialista, com altos custos e possivelmente com viés cognitivo. O *framework* automatiza o processo de construção de conjuntos de treinamento com rótulos fracos, treinamento de modelos e inferência de rótulos finais.

A abordagem central do *Snorkel* é baseada em programação de modelo (*model programming*), onde os usuários escrevem funções de geração de rótulos (*labeling functions*) que atribuem rótulos fracos às instâncias de treinamento. Essas funções podem ser heurísticas, regras ou modelos pré-existentes. Em seguida, o Snorkel utiliza técnicas de aprendizado estatístico para modelar e combinar os rótulos fracos, a fim de obter rótulos mais confiáveis.

Dessa maneira, o *Snorkel* oferece uma interface de programação simples e intuitiva para criar e gerenciar *pipelines* de treinamento com rótulos fracos. Ele também fornece uma variedade de recursos, como algoritmos de aprendizado ativo, técnicas de avaliação de desempenho e ferramentas de visualização, para ajudar os usuários a iterar e aprimorar seus modelos.

Ao permitir o uso de rótulos fracos, o *Snorkel* torna mais fácil e eficiente a construção de conjuntos de treinamento de larga escala, mesmo quando rótulos precisos não estão disponíveis. Isso é especialmente útil em casos de uso onde a anotação manual é cara, demorada ou difícil de obter. O *Snorkel* tem sido aplicado com sucesso em várias áreas, como processamento de linguagem natural, visão computacional, bioinformática e análise de dados em geral.

Nesse contexto, a supervisão programática fraca pode ser útil para melhorar o entendimento da rotução e reduzir o viés. Ao contrário da maioria dos modelos de *machine learning* (ML), que não apresentam visibilidade das camadas internas, em supervisão fraca, os dados de treinamento são gerados a partir do código escrito para esse propósito. Como resultado, obtem-se um maior controle sobre como os rótulos são criados. Por exemplo, é possível observar quais funções de rotulagem contribuem e como elas estão sendo combinadas ao usar supervisão fraca programática para treinar e criar um modelo de ML (28).

Diante disso, destaca-se que a capacidade de interpretação oferece oportunidades para a identificação e a gestão dos vieses no conjunto de dados, bem como na produção do conhecimento gerado durante a implantação desses dados em produção. Atualmente, emprega-se o *Framework Snorkel* em diversas aplicações para indústria, medicina e academia, incluindo, dentre elas, a classificação de produtos e a inicialização de agentes de conversação.

Além disso, o *Snorkel* tem contribuído de maneira destacada para tarefas que envolvam o processamento de linguagem natural (PNL), sendo utilizado, por exemplo, para construir um banco de dados de novas associações genéticas usando documentos científicos e relações químicas entre doenças, resultando na identificação de novos conhecimentos científicos (28).

Ainda no tocante às aplicações para a medicina, nos registros eletrônicos de saúde, as anotações dos

pacientes são uma importante fonte de informações para a IA nos cuidados de saúde. O *Snorkel* tem sido empregado para extrair rótulos de resultados para imagens médicas e melhorar a rotulagem de relatórios de radiologia, monitorar a segurança de dispositivos médicos, classificar fatores de risco e monitorar continuamente os sintomas de COVID-19, apoiados por técnicas de reconhecimento de entidade nomeada (NER) (28, 30, 31).

2.4.1 Programação de Dados

A programação de dados (*data programming*) é um novo paradigma que permite a geração de um grande conjunto de conjuntos de dados de treinamento rotulados programaticamente. Na programação de dados, um usuário cria um conjunto de funções programáveis simples chamadas funções de rotulagem (*labeling functions*) que são usadas para rotular os dados. Cada uma dessas funções de rotulagem fornece um rótulo para cada exemplo de treinamento, ou se abstém. Ao executar várias funções de rotulagem, obtemos informações úteis, mas potencialmente conflitantes, sobre o rótulo de cada exemplo. A programação de dados nos permite agregar esses votos em uma distribuição de probabilidade coerente sobre os rótulos verdadeiros e desconhecidos (28, 30).

Indicadores recentes a respeito de IA demonstraram que, fundamentalmente, os algoritmos de ML não evoluíram de forma radical; o que mudou é a quantidade de dados utilzados por eles. Os dados são empregados em um volume muito maior e têm rótulos melhores. Atualmente, a maioria das soluções de ML tende a ser uma solução completa mais abrangente, que requer ainda mais dados rotulados para funcionar. Deste modo, uma maneira de escalar-se a solução para esse entreve é com a programação de dados, que coopera para criar dados de treinamento programaticamente. O código que permite isso é chamado de função de rotulagem; as quais criam rótulos fracos que permite a utilização da supervisão fraca.

A programação de dados tem muitas vantagens, como a codificação do conhecimento do domínio, representando restrições e situações da vida real de forma reutilizável e atualizável, em vez de rótulos de treinamento individuais. Permite ainda que seja incorporado simultaneamente uma ampla gama de recursos de supervisão para treinamento de maneira baseada em princípios, além de reduzir o tempo e o esforço necessários para implantar novos modelos (28).

2.4.2 Arquitetura do Snorkel

O *workflow* do Snorkel envolve as seguintes etapas para construir e treinar modelos usando rótulos fracos, conforme a figura 2.6.

- **Definição do problema**: Identificar o problema de aprendizado de máquina para resolver e determinar se a supervisão fraca é aplicável, estabelecendo o objetivo do modelo e o tipo de rótulos fracos disponíveis.
- **Criação das funções de rotulação (*labeling functions*)**: Escrever as funções de rotulação que atribuam rótulos fracos às instâncias de treinamento. Essas funções podem ser heurísticas, regras programáticas ou modelos pré-existentes. O objetivo é criar várias funções para diversificar os rótulos e capturar diferentes perspectivas.

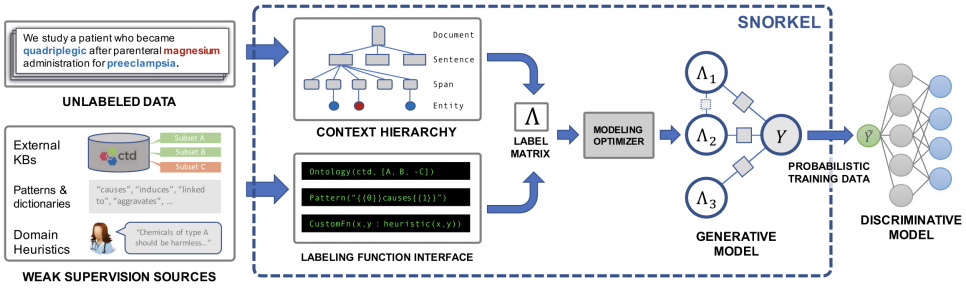

Figura 2.6: *Overview* do Workflow do Snorkel.

- **Criação do conjunto de treinamento com rótulos fracos**: Utilizar as *labeling functions* para rotular um grande conjunto de treinamento com rótulos fracos, sendo que esses rótulos fracos podem conter erros ou serem inconsistentes.
- **Modelagem estatística dos rótulos fracos**: Aplicar técnicas de aprendizado estatístico para modelar e combinar os rótulos fracos, a fim de obter rótulos mais confiáveis.
- **Treinamento do modelo final**: Usar os rótulos estatísticos refinados para treinar um modelo final de aprendizado de máquina. O Snorkel pode ser integrado com várias bibliotecas de aprendizado de máquina, como *TensorFlow*[1], *PyTorch*[2] ou *Scikit-learn*[3], para treinar o modelo final. No presente trabalho foram aplicados os modelos LSTM e BiLSTM.
- **Avaliação e iteração**: Avaliar o desempenho do modelo final usando métricas apropriadas para o problema e, posteriormente, analisar os resultados e iterar no processo, ajustando as *labeling functions* ou adicionando novas, se necessário, para melhorar a qualidade dos rótulos fracos e o desempenho do modelo.

2.4.3 Funções de Rotulação

As funções de rotulação (*labeling functions*) no contexto do *Snorkel* são funções programáticas que atribuem rótulos fracos às instâncias de treinamento. Essas funções são criadas pelos usuários e podem ser heurísticas, regras ou modelos pré-existentes. Elas são projetadas para capturar diferentes perspectivas ou critérios para a atribuição de rótulos, conforme a seguinte relação:

- **Heurísticas simples**: Essas funções aplicam regras simples baseadas em heurísticas para atribuir rótulos. Por exemplo, em um problema de classificação de emails como "spam"ou "não spam", uma função de rotulação pode verificar a presença de palavras-chave específicas (como "oferta", "grátis"ou "ganhe dinheiro") no corpo do email e atribuir o rótulo "spam"se encontrar alguma delas.
- **Modelos pré-existentes**: Pode-se usar modelos de aprendizado de máquina pré-treinados ou modelos auxiliares como funções de rotulação. Esses modelos podem ter sido treinados anteriormente em tarefas relacionadas ou em conjuntos de dados diferentes.

[1] https://www.tensorflow.org/
[2] https://pytorch.org/
[3] https://scikit-learn.org/stable/

- **Combinação de heurísticas**: Pode-se criar funções de rotulação combinando várias heurísticas ou regras. Essas funções levam em consideração múltiplos critérios para atribuir rótulos. Por exemplo, em um problema de análise de sentimentos em textos, pode-se criar uma função que verifica a presença de palavras positivas e negativas e atribui rótulos baseados na frequência relativa dessas palavras no texto.

- **Aprendizado ativo**: O Snorkel também suporta o uso de aprendizado ativo para criar funções de rotulação. Nesse caso, as instâncias de treinamento são selecionadas para serem rotuladas por um especialista humano. As funções de rotulação podem então usar informações do especialista na temática para atribuir rótulos a outras instâncias não rotuladas. Esse processo iterativo ajuda a melhorar gradualmente a qualidade dos rótulos fracos.

É importante notar que as funções de rotulação podem não ser perfeitas e podem gerar rótulos fracos incorretos ou imprecisos. No entanto, o *Snorkel* utiliza técnicas de modelagem estatística para combinar e inferir rótulos mais confiáveis, mesmo quando as funções de rotulação individuais não são precisas.

2.4.4 Limitações do Framework Snorkel

Embora o *Snorkel* seja uma ferramenta poderosa para lidar com supervisão fraca e construir modelos com rótulos fracos, ele também apresenta algumas limitações. Aqui estão algumas das limitações comuns do *Snorkel*:

- **Dependência de rótulos fracos de alta qualidade**: O desempenho do *Snorkel* depende da qualidade dos rótulos fracos gerados pelas *labelling functions*. Se as funções de rotulação não forem bem projetadas ou se os rótulos fracos forem inconsistentes ou incorretos, isso pode afetar negativamente o desempenho do modelo final.

- **Dificuldade em modelar dependências complexas**: O *Snorkel* é mais adequado para problemas em que as dependências entre as funções de rotulação são simples e não há dependências complexas entre os rótulos fracos. Lidar com dependências complexas entre as funções de rotulação pode ser um desafio no Snorkel e pode exigir técnicas adicionais para modelar essas dependências.

- **Limitações computacionais**: O emprego do *Snorkel* pode ser computacionalmente intenso, especialmente quando há um grande número de funções de rotulação ou um grande conjunto de treinamento. A criação de modelos estatísticos para inferir rótulos finais pode exigir recursos computacionais substanciais.

- **Necessidade de iteração e ajuste manual**: O processo de desenvolvimento de modelos no *Snorkel* geralmente requer iteração e ajuste manual das funções de rotulação, bem como das técnicas de modelagem estatística. Isso pode exigir conhecimento especializado e tempo significativo para obter resultados satisfatórios.

- **Rótulos finais ainda podem ser fracos**: Embora o Snorkel melhore a qualidade dos rótulos fracos por meio de técnicas de modelagem estatística, os rótulos finais ainda podem ser considerados fracos

em comparação com rótulos anotados manualmente. Isso pode resultar em uma menor precisão ou confiabilidade do modelo final.

- **Requisito de um grande conjunto de treinamento**: O *Snorkel* pode exigir um grande conjunto de treinamento com rótulos fracos para obter bons resultados. Se apenas um número limitado de rótulos fracos estiver disponível, pode ser desafiador obter melhorias significativas no desempenho do modelo final.

Diante das considerações apresentadas, deve ressaltar a importância das limitações do *Snorkel* e considerar se o *framework* é o mais apropriado para o problema em questão, bem como se há disponibilidade de rótulos fracos. Em certos casos, outras abordagens de aprendizado de máquina supervisionado ou semi-supervisionado podem ser mais adequadas.

2.5 MÉTRICAS DE DESEMPENHO

A identificação de informações estatisticamente de maior relevância, a partir de postagens em redes sociais, é uma tarefa desafiadora em virtude da subjetividade das temáticas analisadas, em especial a pandemia do COVID-19, com os seus antagonismos e complexidade natural. Isso se torna ainda mais difícil, em virtude da escassez de conjuntos de dados para treinamento com rótulos reais no idioma português para a temática proposta.

Nesse sentido, em decorrência da necessidade de um grande *dataset* para treinar os modelos LSTM e BILSTM, associados ao elevado custo para rotular os dados manualmente, pela ação de um especialista, tornou-se indispensável a elaboração de um conjunto de dados de forma programática, a fim de disponibilizar dados rotulados em quantidade suficiente para mitigar um eventual desbalanceamento e validar a eficiência dos modelos de detecção avaliados.

3 TRABALHOS CORRELATOS

Recentemente, as técnicas de *Deep Learning*, têm alcançado resultados significativos em diversas áreas de pesquisa. Nesse contexto, torna-se oportuno aplicar essas técnicas, como as Redes Neurais Recorrentes, especialmente do tipo Long Short-Term Memory (LSTM), no campo do Processamento de Linguagem Natural (NLP) em língua portuguesa, para identificar as informações estatisticamente de maior relevância, provenientes do *microblog* Twitter.

As notícias falsas têm se tornado um tema de grande importância no campo de Processamento de Linguagem Natural devido ao impacto negativo que exercem em nossa sociedade. Apesar de sua relevância, há escassez de conjuntos de dados disponíveis em português brasileiro e, em sua maioria, esses conjuntos são compostos por poucas amostras (32).

No âmbito das pesquisas relecionadas, a disponibilidade de conjuntos de dados rotulados pode ser considerado um dos principais óbices para a automatização da detecção de notícias falsas. Existem alguns conjuntos de dados a respeito de *fake news* e campanhas de desinformação, mas no idioma português, o trabalho ainda é incipiente, havendo a oportunidade de desenvolver um conjunto de dados para treinar algoritmos de machine learning (32).

Em (32), foi elaborado um novo conjunto de dados de notícias falsas chamado *FakeRecogna*, que conta com um número significativamente maior de amostras, incluindo notícias atualizadas e abrangendo diversas categorias importantes. Para avaliar o conjunto de dados criado, foram utilizados classificadores tradicionais, tais como Naive Bayes, Optimum-Path Forest e Support Vector Machines. Além disso, uma Rede Neural Convolucional também foi empregada para detectar notícias falsas no conjunto de dados proposto.

A partir desse desafio, foi empregado no presente estudo o *framework Snorkel*, (31), que permite a construção de conjuntos de dados de treinamento de forma programática, mitigando os custos de empregar recursos especializados, como no caso de jornalistas atuando como *fact checkers*.

Diversas técnicas de aprendizado de máquina vêm sendo empregadas para a detecção de campanhas de desinformação. Os estudos para o processo de detecção de *fake news* são baseados em pesquisas teóricas (33, 34) e práticas (35, 36, 37), integrando uma grande variedade de disciplinas, como a linguística e a ciência da computação (6).

Os sistemas de detecção automática de *fake news* são baseados em métodos de classificação de textos (6) e apresentam vantagens sobre o processo realizado de forma manual, pois os julgadores humanos são propensos a preconceitos. Ademais, julgamentos que dependam de várias fontes de informação, como vídeos, áudios e outras mídias podem sobrecarregar o julgador e levar a atrasos e erros.

Em (38), é apresentada uma abordagem simples para detectar notícias falsas usando o classificador *Naive Bayes*. Essa estratégia foi implementada como um sistema de software e testada em um conjunto de dados composto por postagens de notícias do *Facebook*[1]. Os resultados obtidos revelaram uma precisão de

[1] https://www.facebook.com/

classificação de aproximadamente 74% no conjunto de teste, o que é considerado um resultado satisfatório, considerando a simplicidade do modelo utilizado.

Em (39), foram avaliados o desempenho de cinco modelos de aprendizado de máquina (*Logistic regression, Decision Tree, K-Nearest Neighbor (KNN), Random Forest, Support Vector Machine*) e três modelos de aprendizado profundo (*Convolutional Neural Networks, LSTM, Gated Recurrent Unit*) em dois conjuntos de dados distintos, contendo notícias falsas e verdadeiras, com utilização de validação cruzada. Adicionalmente, foram empregadas técnicas de representação de texto, como frequência de termo, frequência de documento inversa de frequência de termo e técnicas de incorporação para os modelos de aprendizado de máquina e aprendizado profundo, respectivamente.

Embora os sistemas de detecção automática não possam ser empregados de forma completamente independente, eles podem auxiliar os especialistas humanos nos procedimentos de *fact-checking* (6). Nesse aspecto, a identificação de conteúdo relevante permite que a classificação de texto aponte para alegações exageradas, linguagem excessivamente emocional ou um estilo incomum nas principais fontes de notícias. Desta forma, é possível a detecção de *fake news* com base no estilo, no contexto e no conhecimento de determinado assunto.

A detecção automatizada de campanhas de desinformação pode ser conceituada como a tarefa de avaliar a veracidade das afirmações nas notícias (1), empregando diversas técnicas de Inteligência Artificial, dentre elas *Machine Learning*, *Data Mining* e *Natural Language Processing*.

Nesse escopo, destacam-se as tarefas de classificação de rótulos multiclasses (falso, verdadeiro, parcialmente falso, parcialmente verdadeiro) (40) e regressão (41). Uma das condições para que os sistemas classificadores de notícias falsas alcancem bons desempenhos é que existam dados rotulados suficientes (1), sendo mais comuns os modelos de *Machine Learning* que empregam métodos supervisionados de aprendizagem.

4 STALLA: UM FRAMEWORK PARA ANÁLISE DE FONTES ABERTAS

O *framework* proposto no presente trabalho remete ao acrônimo do idioma inglês, *scraping, transforming, auto-labelling, learning and analysis* (Figura 4.1). O *framework* STALLA contempla em sua nomenclatura as etapas de procedimentos técnicos necessários para a obtenção desde a coleta dos dados até análise e produção do conhecimento, alcançando a extremidade do "funil"da Inteligência. A principal intenção do *framework* e amplificar a produtividade dos trabalhos atinentes à Atividade de Inteligência governamental.

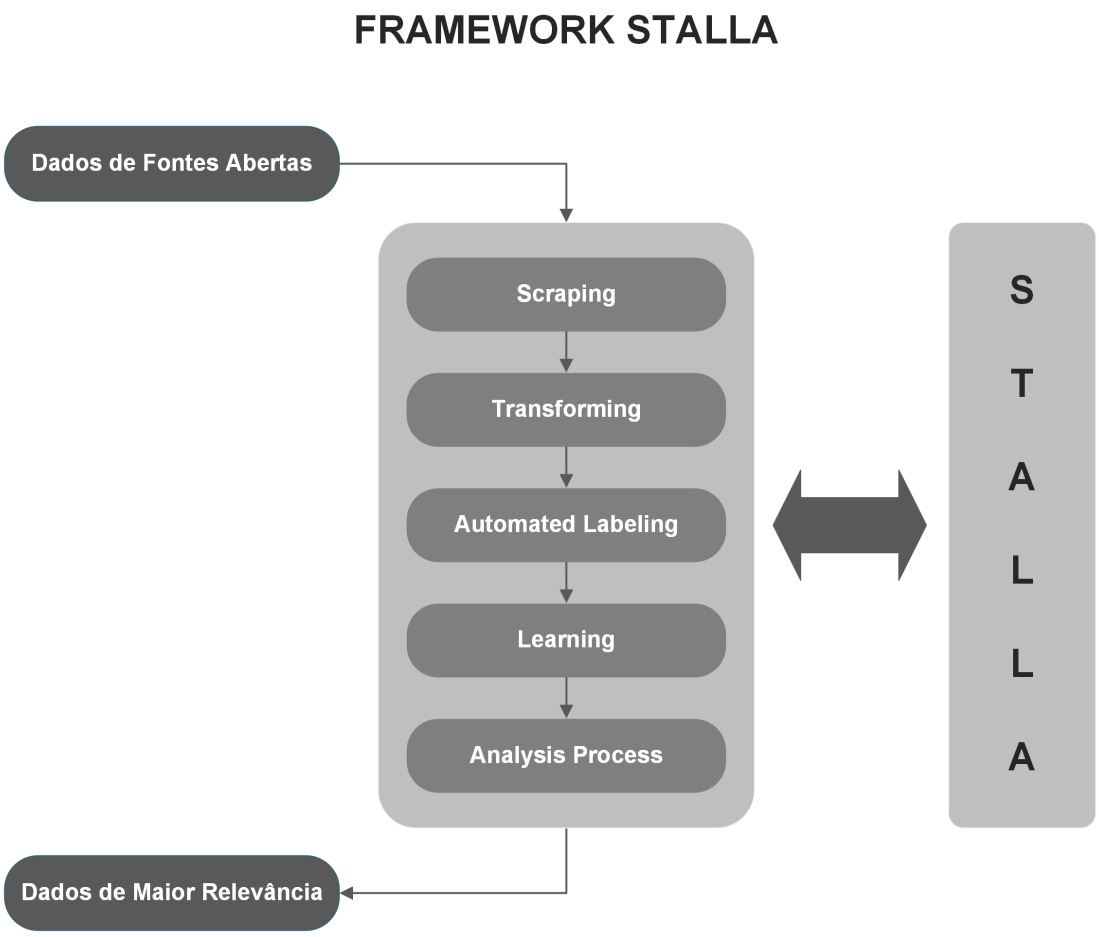

Figura 4.1: Framework STALLA

4.1 INTELIGÊNCIA DE FONTES ABERTAS

Na Internet a informação é abundante e disponível. A informação aberta pode ser usada para extrair insights, analisar tendências e tomar decisões. A grosso modo a disciplina de Inteligência de Fontes Abertas

(OSINT) refere-se à coleta, análise e interpretação de informações publicamente disponíveis de diversas fontes, envolvendo a extração de inteligência acionável de fontes abertas, como mídias sociais, artigos de notícias, sites, registros públicos, dentre outros. No tocante à Inteligência de Fontes Abertas vale à pena ressaltar as suas seguintes peculiaridades:

- Deve concentrar-se em informações acessíveis a qualquer pessoa, sem violar a privacidade ou recorrer a meios ilegais;
- Possuir uma abordagem multidisciplinar que integre várias disciplinas, incluindo análise de dados, pesquisa, tecnologia e pensamento crítico;
- Utilizar uma variedade de métodos e técnicas para coletar e analisar informações de forma eficaz.

Cabe destacar que cada rede social possui suas peculiaridades. O Facebook[1], como a maior plataforma de mídia social, atualmente, com seus 2,96 bilhões de usuários, oferece uma ampla gama de recursos para se conectar com amigos, compartilhar conteúdo e ingressar em comunidades.

O WeChat[2], bastante popular na China, combina redes sociais, mensagens e comércio eletrônico em um aplicativo, permitindo que os usuários paguem contas, marquem compromissos e muito mais. O Instagram[3] concentra-se em conteúdo visual, com forte ênfase em fotos e vídeos, o que o torna ideal para mostrar criatividade e narrativa visual.

O TikTok[4] ganhou imensa popularidade para vídeos curtos, oferecendo uma plataforma altamente envolvente e divertida. O Snapchat[5] se diferencia com conteúdo que desaparece, filtros de realidade aumentada e uma forte ênfase em mensagens privadas.

O Telegram[6] é conhecido por seus recursos de privacidade e segurança, oferecendo mensagens criptografadas, mensagens autodestrutivas e bate-papos secretos. O Twitter[7] é especializado em atualizações concisas e em tempo real, tornando-o uma plataforma para as últimas notícias, tendências e conversas. O Pinterest[8] se destaca como uma plataforma de descoberta visual, permitindo que os usuários encontrem e salvem inspiração, receitas, moda e muito mais por meio de pins personalizados.

Por fim, o WhatsApp[9] concentra-se em mensagens privadas, com criptografia de ponta a ponta, chamadas de voz e vídeo e bate-papos em grupo, tornando-o popular para comunicação pessoal e profissional. Quando busca-se obter dados a partir de OSINT é de suma importância entender a fonte, o tipo de dado público e a relevância da fonte quanto à temática que se investiga. Na figura 4.2 é apresentado o ranking, em 2023, das redes sociais mais populares populares por milhões de usuários ativos, segundo a Statista[10], plataforma online alemã especializada em coleta e visualização de dados.

[1] https://www.facebook.com/
[2] https://www.wechat.com/pt/
[3] https://www.instagram.com/
[4] https://www.tiktok.com/pt-BR/
[5] https://www.snapchat.com/pt-BR
[6] https://web.telegram.org/
[7] https://twitter.com/home
[8] https://br.pinterest.com/
[9] https://www.whatsapp.com/
[10] https://www.statista.com/

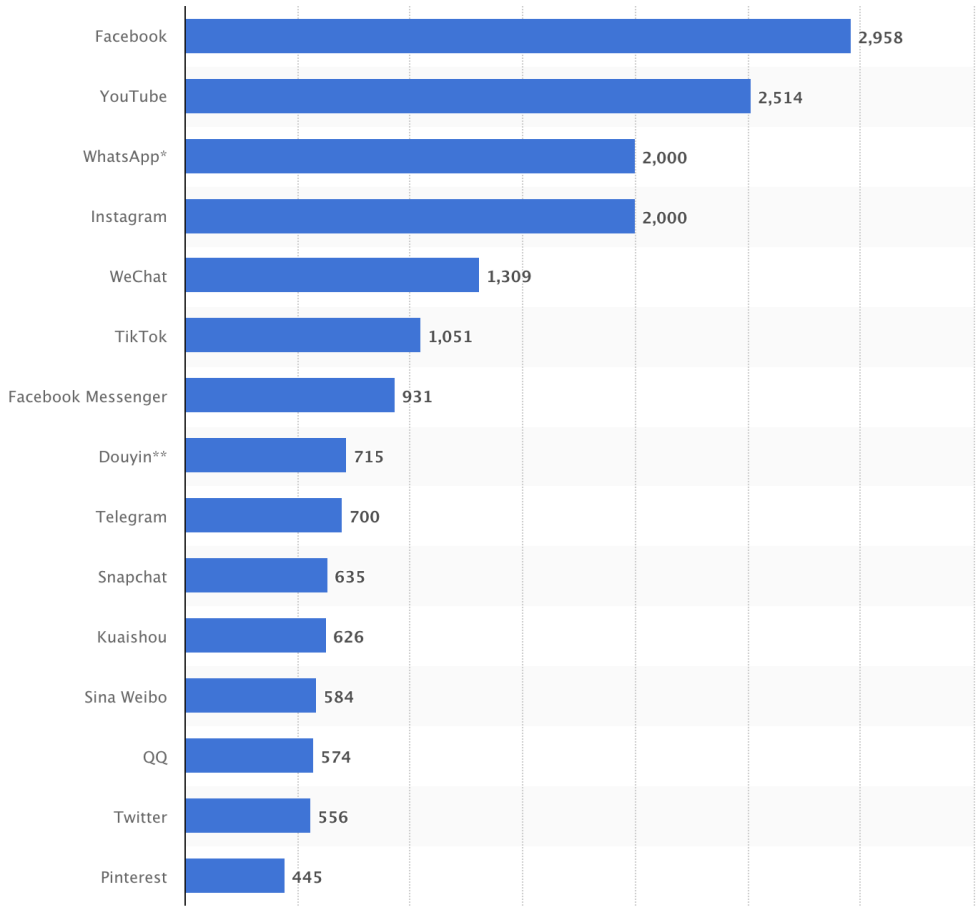

Figura 4.2: Ranking das redes sociais mais populares em 2023.

Assim, os dados oriundos de OSINT podem possuir aplicação em inúmeros domínios: suporte à inteligência de ameaças, contraterrorismo, investigações criminais e identificação de riscos potenciais à segurança pública, inteligência competitiva, pesquisa de mercado, monitoramento de marca, identificação de tendências emergentes ou preferências de consumidores, identificação de vulnerabilidades, monitoramento de ameaças online e *footprint* digital de organizações ou indivíduos.

4.2 PROCESSAMENTO DA INFORMAÇÃO

O processamento de informações para criar conhecimento de Inteligência envolve várias etapas e considerações importantes. Em primeiro lugar, os analistas precisam coletar informações de fontes confiáveis. Isso pode envolver a realização de pesquisas completas, consulta a fontes confiáveis e análise de dados relevantes. Ao garantir a precisão e a credibilidade das informações, os analistas podem estabelecer uma base sólida para seu conteúdo.

Uma vez coletadas as informações, a etapa seguinte é a análise e interpretação. Isso envolve procurar padrões, tendências e insights nos dados. Para isso, os analistas devem avaliar criticamente as informações, identificar pontos-chave e estabelecer conexões entre diferentes partes dos dados e informações. Esse

processo analítico ajuda a moldar a narrativa e o direcionamento do conhecimento, garantindo que ele seja assertivo e relevante.

A partir do entendimento do quão complexo, árduo e amplo é o trabalho de processamento realizado pelos analistas, o *framework* proposto traz uma metodologia para as tarefas otimizar a produção do conhecimento. Desta forma, durante a fase de processamento, realiza-se a carga dos dados brutos obtidos das fontes abertas em um banco de dados temporário para preparação, limpeza, tabulação e demais transformações necessárias. Ainda nessa fase, são elaboradas as funções de rotulação, tendo por base o conhecimento de especialistas na témática a ser analisada. Com base nessas funções, desenvolve-se um *dataset* temático orientado ao estudo de caso proposto.

A exploração dos dados é uma tarefa acessória, cujo principal objetivo é enriquecer o quadro de referência do analista de Inteligência para conceber as funções de rotulação, o mais assertivamente possível. Para isso, emprega-se técnicas de *analytics* com o propósito de visualizar os dados e as suas métricas, como também deduzir os correlacionamentos entre as entidades, resultando em diagramas, gráficos e nuvens de palavras.

4.3 PRODUÇÃO DO CONHECIMENTO

As principais agências de Inteligência dos Estados nacionais utilizam uma metodologia (42) que relaciona dados, informação e produção do conhecimento (Figura 4.3). Essa metologia permite identificar as tarefas precípuas para a produção do conhecimento de Inteligência, contribuindo para a modelagem do *framework* proposto nesse trabalho.

A metodologia destaca a importância do emprego de grandes quantidades de dados para o processamento e a obtenção de informações significativas, visando a produção do conhecimento e, consequentemente, o acessoramento do poder decisório nacional. Nesse sentido, foram levantados os procedimentos técnicos que representam as etapas de coleta; processamento e exploração; e análise e produção do conhecimento, afim de replicar o conceito metodológico com os dados oriundos de OSINT.

4.4 FASES DO CICLO DE PRODUÇÃO DO CONHECIMENTO

As fases do Ciclo de Produção do Conhecimento de Inteligência, sao estrtuturadas em quatro etapas, sendo consenso na maioria das metodologias das agências de Inteligência de Estado dos países (42). As fases são: Planejamento e Direção, Coleta, Análise e Disseminação. Dentro deste paradigma, a seguir será explicitada os procedimentos de cada fase e sua respectiva equivalência no *framework* proposto.

4.4.1 Fase de Planejamento e Direção

A fase de Planejamento e Direção do Ciclo de Inteligência é responsável por identificar as fontes de dados relevantes e os métodos de coleta de informações. A fase envolve o estabelecimento de uma

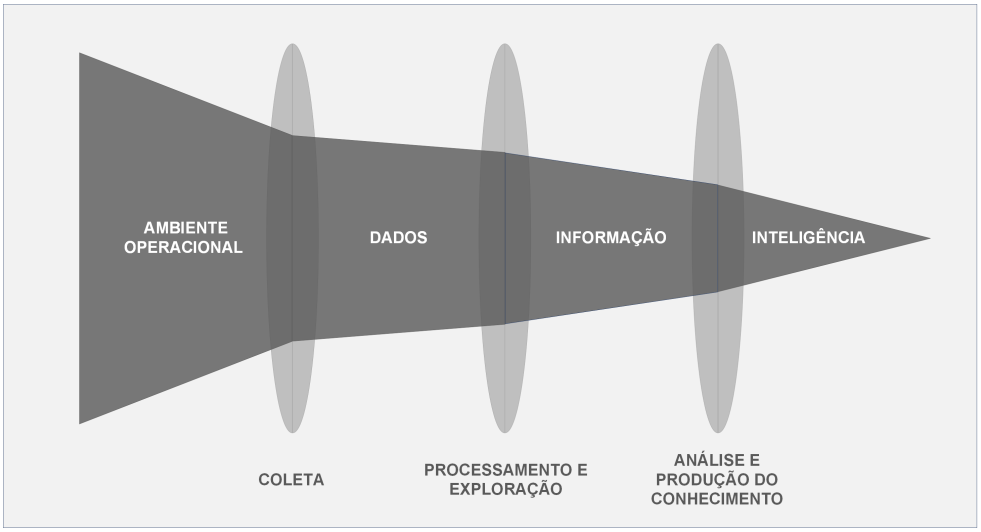

Figura 4.3: Relação entre dados, informação e produção do conhecimento.

estratégia de análise e a definição dos produtos de inteligência que serão gerados. É nessa etapa que ocorre a coordenação das atividades e o direcionamento das ações necessárias para garantir que o ciclo de inteligência siga um caminho eficiente e eficaz.

No escopo do *framework* proposto essa fase é executada de maneira similar, buscando fontes abertas condizentes com o tipo demanda. Existem inúmeras redes sociais, um trabalho que busca a sensibilidade e a relevância sobre uma temática pela população é diferente de um trabalho que busca apenas uma informação precisa de dados numéricos. No caso de uso do STALLA o foco foi em redes sociais, em específico o Twitter[1], pela sua característica de divulgação quase instântanea de idéias, por aproximadamente seus 24 milhões de usuários, no Brasil, conforme dados de 2023.

4.4.2 Fase de Coleta

Na fase de Coleta do Ciclo de Inteligência são obtidos os dados e informações necessárias para o processo de inteligência. São utilizadas diversas fontes, como documentos, relatórios, bancos de dados, entrevistas, mídias sociais, entre outros. A coleta de dados é realizada de acordo com o planejamento estabelecido na fase anterior. É necessário utilizar métodos adequados de coleta, como pesquisas, observação, monitoramento online, entre outros. Durante a fase de Coleta, é preciso organizar e armazenar os dados de forma segura e acessível.

A fase de coleta do *framework* operacionaliza a obtenção dos dados abertas. Para isso existem inúmeras maneiras de realizar o procedimento sendo as mais usuais o *Web Scraping* e o consumo por meio de *Application Programming Interface* (API). No caso de uso do *framework* foi empregado o *Web Scraping* através da biblioteca *Snscrape*[2] o que facilitou o procedimento da coleta de dados por entidades-campos do microblog *Twitter*, como por exemplo: corpo textual do *tweet*, quantidade de curtidas, quantidade de repostagens, quantidade de seguidores de perfis, dentre outras.

[1] https://twitter.com/home
[2] https://github.com/JustAnotherArchivist/snscrape

4.4.3 Fase de Análise

A Análise é a fase em que os dados coletados são transformados em *insights* acionáveis. Nesse momento, as informações são examinadas, interpretadas e contextualizadas para extrair significado e identificar padrões, tendências e relações de causa e efeito. A análise pode envolver o uso de diferentes técnicas e métodos, como estatísticas, modelagem, visualização de dados e análise qualitativa.

No STALLA a fase de Análise é implementada em duas vertentes. A primeira é a obtenção de *insights* para compreensão da temática a partir de ferramentas analíticas, contribuindo para ampliar o quadro de referência do analista e, consequentemente, a elaboração das funções de rotulação que irão compor o processo de treinamento dos dados por meio da supervisão fraca. Para obter esse entendimento a partir dos *tweets* coletados, foram elaboradas tabelas de ranking de relevância e confiabilidade das fontes de informações, bem como nuvens de palavras empregando a biblioteca MatplotLib[1].

Com o entendimento e ampliação do quadro de referência do analista sobre a temática, o trabalho seguinte foi categorizar os dados da maneira de interesse, no experimento buscou-se categorizar os dados em relevantes e não relevantes, para isso foi usada a biblioteca *Snorkel*[2], que auxiliou na elaboração de um modelo de IA capaz de exprimir uma classificação com viés o mais próximo possível de um analista.

4.4.4 Fase de Difusão

A disseminação da produção do conhecimento de Inteligência pode ocorrer por meio de relatórios, apresentações, reuniões, painéis de controle, *dashboards* ou outras formas de comunicação visual e verbal. Nesse sentido, é essencial adaptar o formato e a linguagem para atender às necessidades e capacidades dos destinatários, tornando a informação acessível e compreensível.

O STALLA não contempla um procedimento técnico para a fase de difusão, porém o produto final é algo que pode ser iterado sucessivas vezes a fim de se obter maior assertividade, bem como resultar em um novo direcionamento que retroalimenta o ciclo de produção de conhecimento.

4.5 TÉCNICAS EMPREGADAS PARA COLETA EM FONTES ABERTAS

Os métodos e técnicas para coleta em fontes abertas são os mais variados possíveis, sendo possível citar os seguintes:

- Pesquisa manual *on-line*: realização de pesquisas sistemáticas usando motores de busca, bancos de dados e plataformas de mídia social, etc.
- Monitoramento de mídias sociais: para coletar informações, rastrear tendências e identificar possíveis ameaças ou riscos.
- *Web Scraping*: extração de dados de sites e plataformas online usando ferramentas especializadas ou

[1] https://matplotlib.org/
[2] https://www.snorkel.org/

técnicas de programação, onde um *Web Crawler* é a instância que executa o código de *Web Scraping*.

- Análise geoespacial através de sistemas de informações geográficas (GIS) e imagens de satélite para analisar locais e padrões.
- Análise de Rede e tráfico de rede, mapeamento de conexões e relacionamentos entre indivíduos, organizações ou entidades para identificar padrões ou associações ocultas.
- Mineração de texto e análise de sentimentos, analisando grandes volumes de texto para identificar padrões, sentimentos e tendências.
- Análise de imagem e vídeo, examinando o conteúdo visual para autenticação, reconhecimento de objetos ou identificação de detalhes importantes.

Para o desenvolvimento desses métodos e técnicas foram empregadas ferramentas e implementações tecnológicas que viabilizaram a execução do trabalho de coleta, as quais podem ser enumeradas a seguir:

- Populares motores de busca: *Google*[1] , *Bing*[2] , *Yandex*[3] e mecanismos de busca especializados permitiram buscas direcionadas e a recuperação de informações relevantes.
- Pesquisa a partir de dados por intermédio de soluções de chatbots que consultam grandes modelos de linguagem (LLMs) de Inteligência Artificial e *Generative Pre-trained Transformer* (GPT), como por exemplo: OpenAI ChatGPT[4], Google Bard AI[5], dentre outros.
- Ferramentas de monitoramento de mídias sociais: plataformas como *Hootsuite*[6], *TweetDeck*[7] e *Brandwatch*[8] que auxiliam na consulta por palavras-chave ou tópicos específicos.
- Ferramentas de raspagem da Web: bilbliotecas Python como *BeautifulSoup*[9], *Requests*[10], *Scrapy*[10] e *Snscrape*[11] que ajudam a extrair dados de sites de forma automática.
- Ferramentas de análise de texto: bibliotecas de processamento de linguagem natural (NLP), como NLTK[12], spaCy[13] e IBM Watson[14], facilitam a mineração de texto, análise de sentimento e compreensão da linguagem.

[1] https://www.google.com.br/
[2] https://www.bing.com/
[3] https://yandex.com/
[4] https://chat.openai.com/
[5] https://bard.google.com/
[6] https://www.hootsuite.com/
[7] https://tweetdeck.twitter.com/
[8] https://www.brandwatch.com/
[9] https://pypi.org/project/beautifulsoup4/
[10] https://pypi.org/project/requests/
[10] https://scrapy.org/
[11] https://github.com/JustAnotherArchivist/snscrape
[12] https://www.nltk.org/
[13] https://spacy.io/
[14] https://www.ibm.com/br-pt/watson

4.6 FRAMEWORK STALLA - ANÁLISE E PRODUÇÃO DO CONHECIMENTO

O processo de Análise, já explicitado, é traduzido pelo STALLA como sucessivas tarefas de normalização, tranformação e rotulação.

4.6.1 Raspagem de Dados

A raspagem de dados (ou scraping no idioma inglês) consiste na obtenção de informações renderizadas na página *html* quando acessada por um *browser*. A depender das tecnologias envolvidas, a complexidade da raspagem pode ser maior, pois alguns sites implementam certos obstáculos para execução do *scraping*. Em contrapartida, existem sites provedores de conteúdo que disponibilizam APIs para facilitar o consumo dos seus dados.

4.6.2 Transformação e Preparação

A transformação e a preparação são procedimentos para padronizar os dados coletados, tendo em vista que as diferentes técnicas de *scraping* podem trazer dados em diversos formatos. Como exemplo, pode-se citar que em uma coleta de dados do tipo texto, é possível que seja necessário remover caracteres especiais e limitar a quantidade de palavras por idéia-chave. No caso da coleta de dados oriundos de imagens, é fundamental empregar técnicas de reconhecimento óptico de carácteres (OCR), para posterior tratamento.

A normalização dos dados pode ocorrer antes e depois do carregamento em um *dataframe*. No caso do carregamento posterior, pode-se utilizar novas colunas para apresentar conteúdos tratados de maneira isolada. Os conteúdos isolados permitem o uso de técnicas de *analytics* para visualizar as informações de diversas formas, como no caso da nuvem de palavras referente ao estudo de caso da Covid-19. A nuvem de palavras como outros artefatos gerados com técnicas de *analytics* são parte do *framework* proposto, pois servem como ferramentas complementares para o desenvolvimento de um quadro de referência sobre uma temática específica (Figura 4.4).

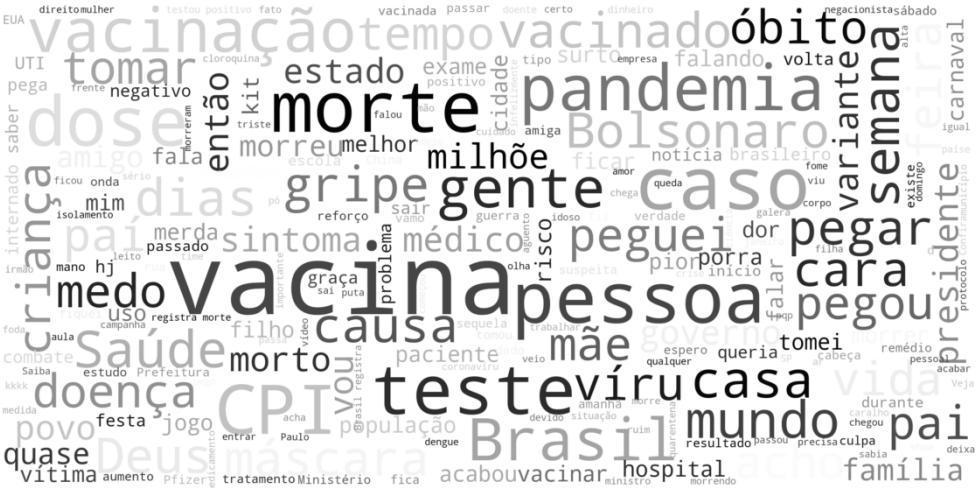

Figura 4.4: Nuvem de palavras gerada a partir do conteúdo do corpo dos tweets.

4.6.3 Rotulação Automatizada

O *automated labelling* consiste na tarefa de classificar, por meio de supervisão fraca, a massa de dados coletada, resultando em um *dataset*, de acordo com o quadro de referência dos especialistas da temática tratada, que no caso do presente estudo, foi a pandemia da Covid-19.

4.6.4 Análise e Produção

As últimas duas fases do *framework* STALLA são o aprendizado de máquina e a análise, as quais são viáveis em decorrência do *dataset* temático desenvolvido, na fase anterior, por meio da rotulação automática (*weak supervision*). Neste ponto, são reunidos os requisitos indispensáveis para identificar as informações mais relevantes para o analista de Inteligência, no contexto da pandemia de Covid-19.

Dessa maneira, considerando-se estudos correlatos que empregaram *deep learning* para análises textuais-semânticas, para desempenhar as fases finais do framework, optou-se pela utilização das *Recurrent Neural Networks* (RNNs) devido a sua vocação no reconhecimento de padrões em dados textuais.

Portanto é evidente o quanto o trabalho de produção do conhecimento de inteligência é muito dispendioso, uma vez que demanda um estudo analítico de grandes quantidades de informação, fenômeno conhecido como *"information overload"*, além do quadro de referência do analista de inteligência, para se obter o conhecimento de interesse (8).

Como consolidação do trabalho experimental foi possível identificar como o Framework STALLA é desenvolvido através de sua visão geral aplicada neste caso de estudo da pandemia do Covid-19, Figura 4.5.

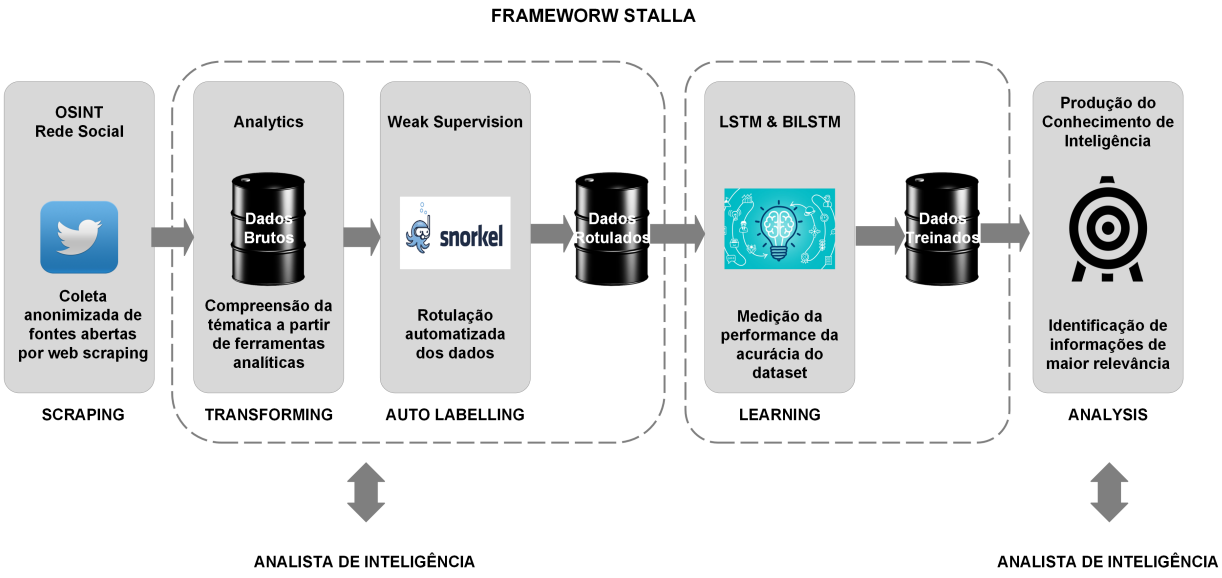

Figura 4.5: Visão geral das funções do FRAMEWORK STALLA.

4.7 EXPERIMENTOS

4.7.1 Configuração dos Experimentos

Os dados utilizados nessa pesquisa são provenientes de uma coleta na rede social Twitter, em virtude desse *microblog* possuir um número abundante de fontes de informação sobre a temática da Covid-19. O procedimento de coleta consiste no uso de *web crawlers*, instâncias de máquinas virtuais em nuvem, que executam um código para realização de *web scraping*.

A técnica de *web scraping* foi executada através de *scripts* desenvolvidos na linguagem de programação *Python*, que utilizam funções das bibliotecas públicas *Snscrape*[1] e *Requests*[2] (43). Na Tabela 4.1 encontram-se as características da referida coleta.

Tabela 4.1: Qualificação do processo de obtenção de dados.

Fonte de OSINT	Rede Social Twitter
Idioma	Português
Quadro de Referência	*Tweets* contendo a palavra "covid" (incluindo variações de maiúsculas e minúsculas)
Período de coleta	Maio de 2021 a Maio de 2022
Amostragem	Aproximadamente 125k *tweets*/mês
Qtd de web crawlers	02 com 25 *threads* de execução cada
Tempo de execução	120 min
Campos coletados	nome de usuário, quantidade de seguidores, localização, quantidade de curtidas, quantidade de retweets, quantidade de réplicas, data da publicação, dispositivo de origem e corpo textual
Qtd de total tweets coletados	1.525.775
MD5 Hash	*c6b5ebc0cb30974fe7eeb8332adee612*
Disponível em	https://shorturl.at/dGKR8

Os dados coletados foram carregados em um *dataframe Pandas*[3], sendo cada coluna preenchida por um campo de interesse, a saber: nome de usuário, quantidade de seguidores, local de origem da postagem (Figura 4.6), quantidade de *likes* (Tabela 4.2), quantidade de réplicas (Tabela 4.3), quantidade de *retweets* (Tabela 4.4), data da postagem, dispositivo de origem da postagem e corpo textual do *tweet*.

Ainda sobre os dados referentes à pandemia da Covid-19, os quais foram obtidos da plataforma Twitter, destaca-se que na maioria das vezes essas informações não estão disponíveis no *tweet*, uma vez que isso depende das políticas de privacidade selecionadas pelo usuário.

Outro ponto a ressaltar, é a politização da temática relacionada à pandemia do Covid-19, o que pode ser

[1] https://github.com/JustAnotherArchivist/snscrape
[2] https://pypi.org/project/requests/
[3] https://pandas.pydata.org/

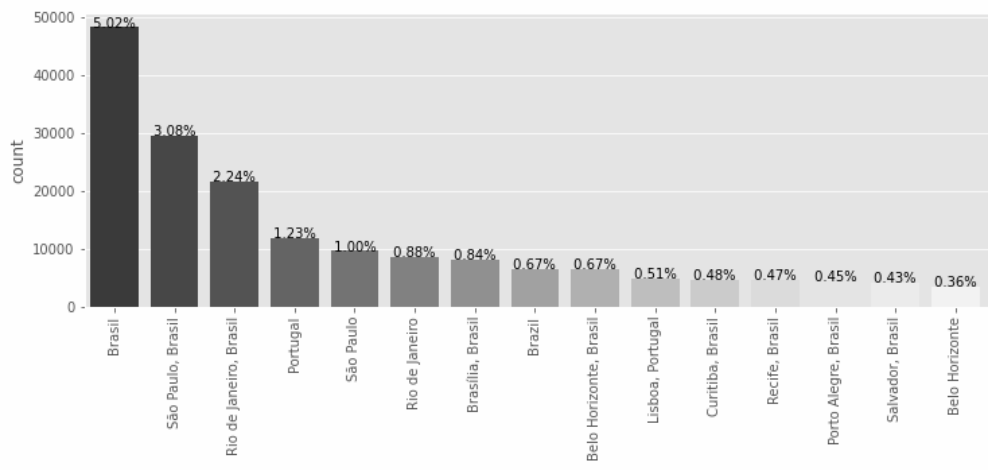

Figura 4.6: Local de origem dos *tweets*.

observado pelo emprego de bibliotecas *Python WordCloud*[1] e *Matplotlib*[2] que resultaram em visualizações que contribuiram para ampliar o quadro de referência do analista de inteligência, favorecendo a elaboração das funções de rotulação.

Tabela 4.2: Top 3 de *tweets* com o maior número de *likes*.

Usuário	**Nr Likes**	**Corpo textual do tweet**
Phenrique1201	177025	“enfim zeramos o cti covid”
aliceeleroy	170440	“vim fazer teste de covid e a moca "com qual cor vc se identifica?"eu: verde caras, as opcoes eram branca/parda/negra”
predonisona	152959	“e agora sera que esses sintomas sao de rinite sinusite h1n1 h3n2 covid h2o ou hb20 hbo max”

Tabela 4.3: Top 3 de *tweets* com o maior número de réplicas.

Usuário	**Nr Réplicas**	**Corpo textual do tweet**
pdrbnt	5391	“vc conseguiu tomar a vacina antes de contrair covid?”
gen_heleno	4224	“impressionante a parcialidade da cnn ao comentar a cpi da covid. algo constrangedor. jornalista mona lisa, dany lima e alguns colegas deviam se unir aos seis senadores da oposicao para se postarem contra o governo. garanto que a audiencia ja vem punindo essa lamentavel postura.”
RomeuZema	3571	“entre os internados com covid em minas hoje, os que nao se vacinaram sao proporcionalmente tres vezes mais do que aqueles que tomaram todas as doses. o problema nao esta na vacina e sim na desinformacao. vacinas salvam vidas! vacine.”

[1]https://pypi.org/project/wordcloud/
[2]https://matplotlib.org/

Tabela 4.4: Top 3 de *tweets* com o maior número de *retweets*.

Usuário	Nr Retweets	Corpo textual do tweet
predonisona	36203	“e agora sera que esses sintomas sao de rinite sinusite h1n1 h3n2 covid h2o ou hb20 hbo max”
KriskaCarvalho	11862	“ele imitou uma pessoa morrendo sem ar com covid, ele riu, ele debochou de alguem sem ar num leito... morreram quase 700 mil ou mais... ele riu, nunca esqueco!”
vozdaresist	11714	“se fizermos um minuto de silencio para cada pessoa morta por covid-19, ficariamos calados por oito anos. imagem da manifestacao no rio de janeiro.”

Para o desenvolvimento do *dataset* sobre os *tweets* de maior valor para o analista foi utilizado o *Snorkel Framework*[1], biblioteca *Python* que permite codificar funções de rotulação para automatizar este processo, no quadro a seguir é apresentado um exemplo de algumas funções de rotulação utilizadas.

```
from snorkel.labeling import labeling_function

@labeling_function()
def lf_14(x):
  return HIGHVALUE if x.Followers > 100 else ABSTAIN

@labeling_function()
def lf_19(x):
  return HIGHVALUE if (x.No_Likes > 5) and (x.No_RT > 15) else ABSTAIN

@labeling_function()
def lf_24(x):
  return LOWVALUE if re.search(r"Android|iPhone",x.Source_device,flags=re.I) and \
  (x.Followers < 20) else ABSTAIN
```

O modelo de aprendizado de máquina para rotulação adotado foi o *Majority Vote*, que é encarregado de reunir os resultados das funções de categorização em um único rótulo. O *Majority Vote* é uma técnica usada para combinar várias funções de rotulagem fracas ou ruidosas, com o objetivo de gerar um rótulo único e mais confiável para um determinado grupo de dados.

O *Majority Vote* funciona através do princípio de agregar as previsões de várias funções de rotulagem para mitigar erros e aumentar a precisão geral do processo de rotulação. A vantagem do *Majority Vote* reside em sua capacidade de alavancar um ’conhecimento coletivo’ de várias funções de rotulagem fracas, mesmo que individualmente tenham precisão limitada. Assim, ao combinar suas saídas, o *Snorkel* pode produzir rótulos mais confiáveis e permitir a criação de conjuntos de dados de treinamento em larga escala para modelos de aprendizado de máquina, como foi o caso da aplicação no STALLA.

No presente trabalho, as funções de rotulação foram desenvolvidas baseadas nas seguintes métricas e combinações das mesmas: a quantidade de seguidores dos autores dos *tweets*, a quantidade de curtidas dos

[1] https://www.snorkel.org/

tweets, a quantidade de *retweets* de cada *tweets*, a quantidade de réplicas de cada *tweets*, o dispositivo ou plataforma de publicação, o local de publicação, os pronomes do corpo textual e o quadro de referência do analista. O quadro de referência do analista foi elaborado a partir da lista de palavras consideradas importantes que podem ajudar obter inferências pertinentes. A seguir temos um exemplo, dessas listas.

```
imprensa = ["UOLNoticias", "UOL", "folha", "CNNBrasil", "YouTube", \
            "JornalOGlobo", "Estadao", "JovemPanNews", "GoogleNews", \
            "revistaoeste", "GloboNews", "geglobo", "SigaGazetaBR", \
            "VEJA", "brasil247", "MPF", "jornalnacional", "gzhdigital", \
            "tvglobo", "bbcbrasil", "estadao", "TwitterBrasil", \
            "revistaforum", "OGloboPolitica", "radioitatiaia", "secomvc", \
            "exame", "gazetadopovo", "RevistaISTOE", "YahooBr", \
            "portalR7", "OGlobo", "agoranoticiasbr", "elpais", \
            "conexaopolitica", "ESPNBrasil", "JornalExtra", "SICNoticias", \
            "DiarioPE", "opovo", "congressoemfoco", "JornalBSM", \
            "CBNoficial", "correio24horas", "YouTubeBrasil", "tercalivre", \
            "g1rio", "NoticiasdaTV", "Twitter"]


remedios = ["pfizer", "cloroquina", "kitcovid", "janssen", "astrazeneca", \
            "oxford", "coronavac", "sputinik", "sputnik", "covaxin", \
            "ivermectina", "johnson"]


entidades = ["anvisa", "minsaude", "govbr", "butantanoficial", \
             "TCUoficial", "fiocruz", "SenadoFederal", "@STF", \
             "governosp", "prefeiturabelem", "DefesaGovBr"]

```

5 AVALIAÇÃO DE DESEMPENHO

Este capítulo apresenta a validação do *Framework* STALLA a partir da comparação entre as implementações das arquiteturas *Long Short-Term Memory* (LSTM) e *Bidirectional Long Short-Term Memory* (Bi-LSTM) para a identificação das informações estatisticamente de maior relevância e que geram maior engajamento. Para isso, apresentaremos a seguir a configuração dos experimentos, bem como a contextualização e a discussão dos resultados.

5.1 CONFIGURAÇÃO DOS EXPERIMENTOS

Para avaliar o modelo proposto foram adotadas as arquiteturas *Long Short-Term Memory* (LSTM) e *Bidirectional Long Short-Term Memory* (Bi-LSTM), comparando-se o resultado proveniente dessas duas implementações. Para o sequenciamento de dados e a divisão do processamento, os dados foram convertidos em vetores tokenizados, com valores de rótulos representativos. A estrutura da divisão do *shape* de treinamento com a biblioteca *Scikit-learn*[1] foi a seguinte:

$X_{train} = 360018$; $X_{test} = 120006$; $Y_{train} = 360018$; e $Y_{test} = 120006$

Dessa maneira, como preparação dos dados, o corpo de texto dos tweets foi tratado com a biblioteca de processamento de linguagem natural *Gensim*[2] que emprega modelos acadêmicos para realizar a tokenização das palavras. Na sequência, foram criados *arrays* por meio da biblioteca *Numpy*[3] para configurar os dados no formato adequado para emprego da rede neural.

Com o *dataset* de treinamento e os corpos dos *tweets*, conjunto de dados de teste, preparados em *arrays*, foi empregada a biblioteca *Keras*[4], que é a implementação de uma interface para customização e desenvolvimento de redes neurais artificiais em *Python*. As características definidas para rede por parâmetros e por configurações de hardware estão descritas na Tabela 5.1.

Tabela 5.1: Parâmetros de treinamento

Variáveis do treinamento	**Descrição**
Plataforma	*Jupyter Notebook / Python 3.9*
GPU	NVIDA GeForce GTX 1060
Optimizer	*RMS prop*
Loss	*Categorical cross-entropy*
Épocas	10
Ativação	*Softmax*

[1]https://scikit-learn.org/stable/
[2]https://pypi.org/project/gensim/
[3]https://numpy.org/
[4]https://keras.io/

Nos experimentos executados foram configurados os seguintes parâmetros: o *Categorical Cross-entropia*, como função de perda devido à unificação dos rótulos verdadeiros em uma única unidade; o popular algoritmo de otimização adaptativo *RMSprop*; a função de ativação *Softmax*, para normalização da saída final da rede e 10 (dez) épocas de treinamento.

O RMSprop, abreviação de *Root Mean Square Propagation*, é um algoritmo de otimização comumente usado em aprendizado de máquina e aprendizado profundo para treinar redes neurais. O algoritmo se apresenta adequado para o caso de uso, pois trata-se de taxas de aprendizado oscilantes, portanto permite uma otimização mais estável e eficiente.

O emprego da função *Loss and Categorical cross-entropy* é típica para tarefas de classificação, particularmente ao lidar com várias classes, como no caso do STALLA onde foram definidas as seguintes classificações para as entidades avaliadas: relevantes, irrelevantes ou a abstenção.

Cada classe é representada como um vetor codificado, onde apenas um elemento é 1 (indicando a verdadeira classe) e todos os outros são 0. A distribuição de probabilidade prevista é obtida da saída de um modelo, geralmente por meio de da função de ativação *Softmax*, gerando um valor de probabilidade para cada classe.

5.2 CONTEXTUALIZAÇÃO E DISCUSSÃO DOS RESULTADOS

A partir das configurações definidas, foram executados os treinamentos levando-se em conta os dois tipos de redes neurais, a LSTM e a Bi-LSTM. A diferença das implementações consiste na característica que na rede bidirecional o fluxo ocorre em ambas direções sendo capaz de utilizar informações de ambos os lados. Na Figura 5.1 pode-se notar uma pequena vantagem no desempenho da rede Bi-LSTM.

Os experimentos indicam ainda que com o aumento da quantidade de dados para treinamento, há uma evolução da acurácia do modelo Bi-LSTM, conforme pode ser constatado na Figura 5.2.

Diante do exposto, verifica-se que para uma quantidade acima de 1 milhão de *tweets* como amostra e tendo por fundamento os critérios tomados para a rotulação do *dataset* de treinamento, foi possível predizer o potencial de um *tweet* recém postado, enquadrar-se a padrões já identificados e, portanto, ser estatisticamente de maior relevância.

Para isso, foi adotado o *F1-score*, que é uma medida de performance da acurácia de um modelo sobre o *dataset*. Cabe destacar que quanto mais próximo de 1, melhor é o modelo de predição.

$$Precision = \frac{VerdadeirosPositivos}{VerdadeirosPositivos + FalsosPositivos}$$

$$Recall = \frac{VerdadeirosPositivos}{VerdadeirosPositivos + FalsosNegativos}$$

$$F1 = \frac{2 * Precision * Recall}{Preciso + Recall}$$

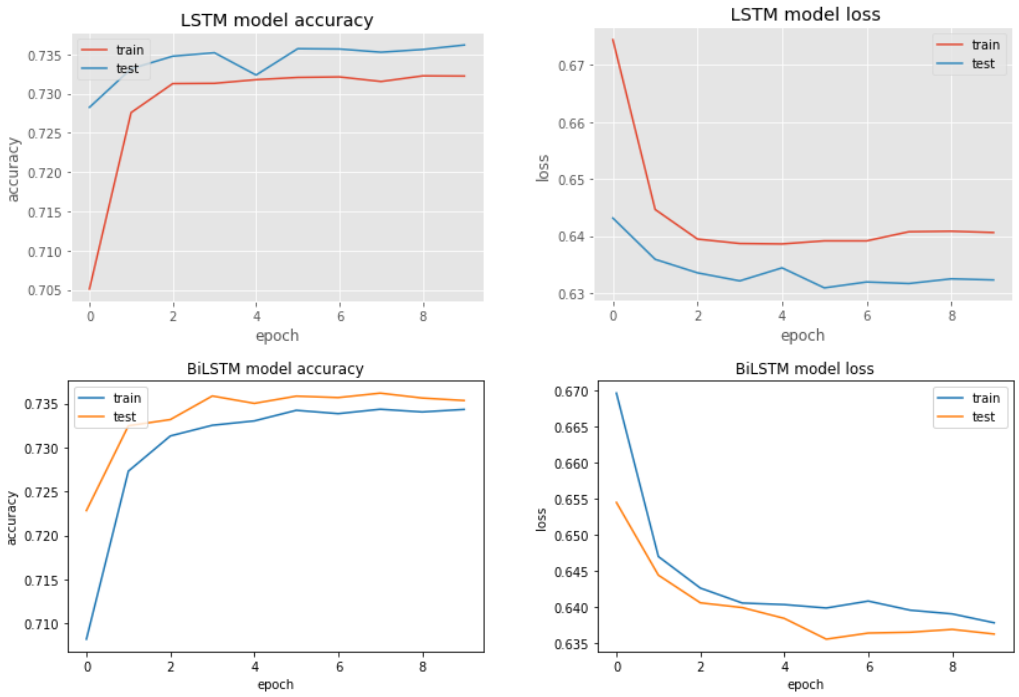

Figura 5.1: Resultado comparativo LSTM X Bi-LSTM.

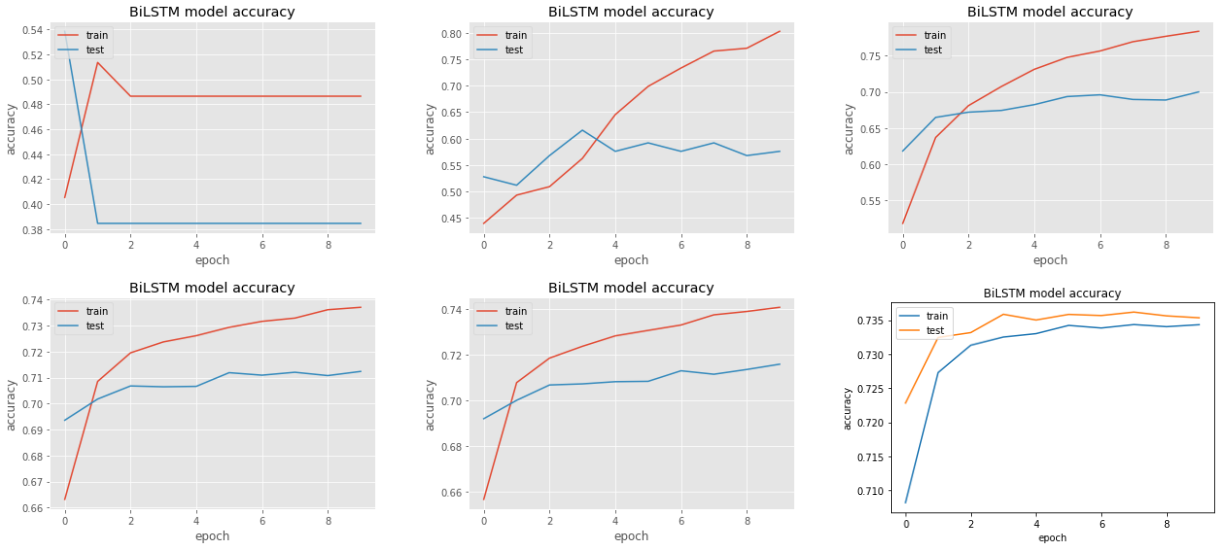

Figura 5.2: Evolução da acurácia frente à quantidade de dados.

Ressalta-se que há contextos em que valores de *F1-score* podem ter interpretações muito específicas. Um exemplo clássico é o modelo de identificação de fraudes bancárias, no qual o valor de 0.95, pode ser um valor extremamente baixo, em função da finalidade a que se propõe, impactando em prejuízos financeiros inadmissíveis. No entanto, nesse estudo de caso, a avaliação da qualidade e da relevância das informações, é uma tarefa extremamente desafiadora, devido à enxurrada de dados produzidos nas fontes abertas, sem o compromisso de prover conhecimento verossímil. Nesta conjuntura, um resultado de *F1-score* superior a $1/2$ representa um valor positivo na análise de conteúdo, conforme indica a Tabela 5.2.

Tabela 5.2: Métricas de desempenho do Framework STALLA.

-*-	**Precision**	**Recall**	**F1-Score**
Accuracy			0.74
Macro Avg	0.49	0.52	0.50
Weighted Avg	0.71	0.74	0.72

6 CONCLUSÃO

A identificação de uma informação que possui potencial para mobilizar usuários em plataformas de redes sociais, aproxima-se da finalidade precípua da Atividade de Inteligência, que é obter o conhecimento em momento oportuno para subsidiar a tomada de decisão pelo gestor responsável. Dessa maneira, o presente trabalho de pesquisa resultou no *Framework* STALLA, o qual permite ao analista de Inteligência alcançar uma maior produtividade na análise de informações provenientes de fontes abertas.

Diante do exposto, conclui-se que o modelo de predição treinado para detectar informações estatisticamente de maior relevância, gerando maior engajamento na plataforma *Twitter*, no contexto da pandemia do Covid-19, atingiu aproximadamente 70% de acurácia, evidenciando a efetividade do *Framework* STALLA na identificação das postagens (*tweets*), potencialmente, mais relevantes, mesmo não se tratando de informações necessariamente verdadeiras.

Por fim, a acurácia de 70% é exclusiva para o caso de estudo que se busca aplicar uma interpretação de valor automatizada com viés de interesse do analista para dados desconhecidos. Em outras temáticas e, a depender, da qualidade das funções de classificação é esperado que o valor da acurácia varie. Cabe ressaltar que para a pandemia da Covid-19, por se tratar de uma temática complexa, apresentando grandes divergências de opiniões, o valor de 70% pode ser considerado satisfatório, se comparado com outras temáticas e contextos.

6.1 TRABALHOS FUTUROS

Como trabalhos futuros planeja-se desenvolver uma tranformação a mais na análise de dados aplicada ao STALLA, obtendo-se vídeos e imagens, a partir de fontes abertas, agregando para esse fim técnicas de reconhecimento de imagem (OCR) e transcrição de áudio (STT).

Nesse sentido, planeja-se, ainda, desenvolver um mecanismo para realização de interpretação semântica com mais precisão, bem como testar o framework nos casos da Guerra da Ucrânia e das eleições presidenciais de 2022, realizadas no Brasil.

REFERÊNCIAS BIBLIOGRÁFICAS

1 OSHIKAWA RAY; QIAN, J. W. W. Y. A survey on natural language processing for fake news detection. In: *Conference on Language Resources and Evaluation (LREC 2020), pages 6086–6093*. [S.l.: s.n.], 2020.

2 SILVA, R. M. e. a. Towards automatically filtering fake news in portuguese. In: *Expert Systems with Applications, v. 146, p. 113199, 2020*. [S.l.: s.n.], 2020.

3 LAZER, D. M. e. a. The science of fake news. In: *Science, v. 359, n. 6380, p. 1094-1096, 2018*. [S.l.: s.n.], 2018.

4 BECHMANN ANJA; NIELBO, K. L. Are we exposed to the same “news” in the news feed? an empirical analysis of filter bubbles as information similarity for danish facebook users. In: *Digital journalism, v. 6, n. 8*. [S.l.: s.n.], 2018. p. 990–1002.

5 KANG CECILIA; GOLDMAN, A. In washington pizzeria attack, fake news brought real guns. In: *New York Times, v. 5, 2016*. [S.l.: s.n.], 2016.

6 ASR FATEMEH; TABOADA, M. T. Big data and quality data for fake news and misinformation detection. In: *Big Data Society, v. 6, n. 1, p. 2053951719843310, 2019*. [S.l.: s.n.], 2019.

7 VOSOUGHI SOROUSH; ROY, D. A. S. The spread of true and false news online. In: *Science, v. 359, n. 6380, p. 1146-1151, 2018*. [S.l.: s.n.], 2018.

8 ALVES, P. M. M. R. O impacto do big data na atividade de inteligência. In: *Revista Brasileira de Inteligência*. [S.l.]: Brasília, DF, n. 13, p. 01-20, dez, 2018, 2018.

9 BRASIL. Decreto nº 8.793, de 29 de junho de 2016. aprova a política nacional de inteligência. In: *Diário Oficial da República Federativa do Brasil, Brasília, DF, Edição 241, 18 dez. 2017. Seção 1, p. 36-39*. [S.l.: s.n.], 2017.

10 ASR, F. T.; TABOADA, M. Big data and quality data for fake news and misinformation detection. *Big Data & Society*, SAGE Publications Sage UK: London, England, v. 6, n. 1, p. 2053951719843310, 2019.

11 WANG, W. Y. "liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*, 2017.

12 THORNE, J.; VLACHOS, A.; CHRISTODOULOPOULOS, C.; MITTAL, A. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 2018.

13 ZHANG, A. X.; RANGANATHAN, A.; METZ, S. E.; APPLING, S.; SEHAT, C. M.; GILMORE, N.; ADAMS, N. B.; VINCENT, E.; LEE, J.; ROBBINS, M. et al. A structured response to misinformation: Defining and annotating credibility indicators in news articles. In: *Companion Proceedings of the The Web Conference 2018*. [S.l.: s.n.], 2018. p. 603–612.

14 FERREIRA, W.; VLACHOS, A. Emergent: a novel data-set for stance classification. In: ACL. *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies*. [S.l.], 2016.

15 ZUBIAGA, A.; LIAKATA, M.; PROCTER, R.; HOI, G. W. S.; TOLMIE, P. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, Public Library of Science San Francisco, CA USA, v. 11, n. 3, p. e0150989, 2016.

16 MITRA, T.; GILBERT, E. Credbank: A large-scale social media corpus with associated credibility annotations. In: *Proceedings of the international AAAI conference on web and social media*. [S.l.: s.n.], 2015. v. 9, n. 1, p. 258–267.

17 GODINHO, A. C.; AMARAL, F. M. do. Decreto nº 8.793, de 29 de junho de 2016. aprova a política nacional de inteligência. *Diário Oficial [da] República Federativa do Brasil*, n. Edição 241, 18 dez. 2017, Seção 1, p. 36–39, 2017.

18 ALVES, P. M. d. M. R. O impacto de big data na atividade de inteligência. *Revista Brasileira de Inteligência*, n. 13, p. 25–44, 2018.

19 AMBROS, C. C.; LODETTI, D. B. Vieses cognitivos na atividade de inteligência: conceitos, categorias e métodos de mitigação. *Revista Brasileira de Inteligência*, n. 14, p. 9–34, 2019.

20 MACHADO, A. M. O impacto de vieses cognitivos sobre a imparcialidade do conteúdo de inteligência. *Revista Brasileira de Inteligência*, n. 13, p. 1–16, 2018.

21 BRASIL. Decreto de 15 de dezembro de 2017-estratégia nacional de inteligência. *Diário Oficial da República Federativa do Brasil*, 2017.

22 HAYKIN, S. S. e. a. *Neural networks and learning machines*. [S.l.]: Upper Saddle River, NJ, USA:: Pearson, 2008.

23 HINTON, G. E. et al. Learning distributed representations of concepts. In: AMHERST, MA. *Proceedings of the eighth annual conference of the cognitive science society*. [S.l.], 1986. v. 1, p. 12.

24 DENG, L.; YU, D. et al. Deep learning: methods and applications. *Foundations and trends® in signal processing*, Now Publishers, Inc., v. 7, n. 3–4, p. 197–387, 2014.

25 RIVAS, P. *Deep Learning for Beginners: A beginner's guide to getting up and running with deep learning from scratch using Python*. [S.l.]: Packt Publishing Ltd, 2020.

26 RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *nature*, Nature Publishing Group UK London, v. 323, n. 6088, p. 533–536, 1986.

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

28 TOK, W. H.; BAHREE, A.; FILIPI, S. *Practical Weak Supervision*. [S.l.]: "O'Reilly Media, Inc.", 2021.

29 BANSAL, A. *Advanced Natural Language Processing with TensorFlow 2: Build effective real-world NLP applications using NER, RNNs, seq2seq models, Transformers, and more*. [S.l.]: Packt Publishing Ltd, 2021.

30 RATNER, A.; BACH, S. H.; EHRENBERG, H.; FRIES, J.; WU, S.; RÉ, C. Snorkel: Rapid training data creation with weak supervision. *The VLDB Journal*, Springer, v. 29, n. 2-3, p. 709–730, 2020.

31 RATNER, A. J. e. a. Snorkel: Fast training set generation for information extraction. In: *Proceedings of the 2017 ACM international conference on management of data. 2017. p. 1683-1686*. [S.l.: s.n.], 2017.

32 GARCIA, G. L.; AFONSO, L. C.; PAPA, J. P. Fakerecogna: A new brazilian corpus for fake news detection. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2022. p. 57–67.

33 DURAN, N. D. e. a. The linguistic correlates of conversational deception: Comparing natural language processing technologies. In: *Applied Psycholinguistics, v. 31, n. 3, p. 439-462, 2010*. [S.l.: s.n.], 2010.

34 HAUCH, V. e. a. Are computers effective lie detectors? a meta-analysis of linguistic cues to deception. In: *Personality and social psychology Review, v. 19, n. 4, p. 307-342, 2015*. [S.l.: s.n.], 2015.

35 APPLING DARREN SCOTT; BRISCOE, E. J. H. C. J. Discriminative models for predicting deception strategies. In: *Proceedings of the 24th International Conference on World Wide Web*. [S.l.: s.n.], 2015. p. 947–952.

36 PéREZ-ROSAS VERóNICA; MIHALCEA, R. Experiments in open domain deception detection. In: *Proceedings of the 2015 conference on empirical methods in natural language processing. 2015. p. 1120-1125*. [S.l.: s.n.], 2015.

37 RUBIN, V. L. e. a. Fake news or truth? using satirical cues to detect potentially misleading news. In: *Proceedings of the second workshop on computational approaches to deception detection. 2016. p. 7-17*. [S.l.: s.n.], 2016.

38 GRANIK, M.; MESYURA, V. Fake news detection using naive bayes classifier. In: IEEE. *2017 IEEE first Ukraine conference on electrical and computer engineering (UKRCON)*. [S.l.], 2017. p. 900–903.

39 JIANG, T.; LI, J. P.; HAQ, A. U.; SABOOR, A.; ALI, A. A novel stacking approach for accurate detection of fake news. *IEEE Access*, IEEE, v. 9, p. 22626–22639, 2021.

40 RASHKIN, H. e. a. Truth of varying shades: Analyzing language in fake news and political fact-checking. In: *Proceedings of the 2017 conference on empirical methods in natural language processing. 2017. p. 2931-2937*. [S.l.: s.n.], 2017.

41 NAKASHOLE NDAPANDULA; MITCHELL, T. Language-aware truth assessment of fact candidates. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2014. p. 1009-1019*. [S.l.: s.n.], 2014.

42 US. Joint intelligence: Joint publication 2-0. In: . Createspace Independent Pub, 2014. ISBN 9781500517366. Disponível em: <https://books.google.com.br/books?id=IK3BoAEACAAJ>.

43 SNSCRAPE. Snscrape: A social networking service scraper in python. In: . GitHub, 2023. Disponível em: <https://github.com/JustAnotherArchivist/snscrape>.

APÊNDICES

A. CÓDIGO FONTE DO CRAWLER DE COLETA DE DADOS

Código fonte:

```
# Importação das bibliotecas necessárias
import snscrape.modules.twitter as sntwitter
import pandas as pd

# Criar lista para indexar todos atributos do Tweet
attributes_container = []

# Execução do scraping para 100 mil Tweets com os atributos definidos
for i,tweet in enumerate(sntwitter.TwitterSearchScraper('covid lang:pt \
since:2021-05-04 until:2022-02-27').get_items()):
    if i>100000:
        break
    attributes_container.append([tweet.user.username, tweet.user.followersCount, \
    tweet.user.location, tweet.likeCount, tweet.retweetCount, tweet.replyCount, \
    tweet.date, tweet.sourceLabel, tweet.content])

# Criação de dataframe com colunas referentes aos atributos de interesse obtidos
tweets_df = pd.DataFrame(attributes_container, columns=["Username", "Followers", \
"Location", "No_Likes", "No_RT", "No_Replies", "Date", "Source_device", "Tweet"])
tweets_df.to_csv(r'100k_tweets_text_about_covid.csv', index=False)
```

B. VISÃO GERAL DO DATASET COLETADO

Index	Username	Followers	Location	No_Likes	No_RT	No_Replies	Date	Source_device	Tweet
0	AZangrande	6579	Itália	1	0	0	2021-05-07 23:59:57+00:00	Twitter for Android	Cantor e compositor Cassiano...mais uma vítima.da covid 19 e desse governo genocida. Partiu aos 77 anos. https://t.co/KV03wYcTQE
1	brunorianoma	897	Santa Catarina, Brasil	0	0	1	2021-05-07 23:59:55+00:00	Twitter Web App	finalmente minha vó vai tomar a vacina do Covid e só passa uma coisa na minha mente: - C L A N D E S T I N O.
2	wesvieirah	1190	Lafayork	2	1	0	2021-05-07 23:59:54+00:00	Twitter Web App	estamos vivendo em um país onde uma pessoa é criticada por se proteger de todas as formas do covid.
3	FanuelArthur	241	NaN	0	0	0	2021-05-07 23:59:52+00:00	Twitter for Android	Se Bolsonaro não tivesse como presidente. Imagina como estaríamos com Haddad? Todo mundo desempregado e mesmo assim várias mortes por covid. Regulação de imprensa e tudo mais.
4	mac_8219	265	SC	0	0	0	2021-05-07 23:59:50+00:00	Twitter Web App	Cassiano, autor da música 'Primavera', morre de Covid no Rio https://t.co/ahICKypBds
5	ErickKayser	2514	Porto Alegre	1	0	0	2021-05-07 23:59:46+00:00	Twitter Web App	Gostei do eufemismo "Novo modelo para conter covid-19", quando na verdade estamos vendo é um "liberou geral", ficando cada um por si, sem nenhuma ação efetiva do Estado. Quando vier a 3º onda, terão ainda a cara de pau de se dizerem surpresos. https://t.co/713ghMA2Dz
6	jessicaximenes	447	NaN	1	0	0	2021-05-07 23:59:44+00:00	Twitter Web App	hoje, faleceu um conhecido meu do interior com 34 anos por causa da Covid. Fazia muitos anos que não o via, mas lembro q sempre q eu ia de férias pra Vargem Grande, eu fazia dindim e deixava pra vender no sacolão da minha tia, ele era meu melhor cliente 😢
7	Vitor__Sa	269	NaN	2	0	0	2021-05-07 23:59:39+00:00	Twitter for Android	Bolsonaro falando merda da china que é o maior parceiro comercial e fornecedor de matéria prima da vacina da covid 19... #JornalNacional #Jn
8	crf_hugoo	297	Santíssimo, Rio de Janeiro	0	0	1	2021-05-07 23:59:36+00:00	Twitter for iPhone	@jessica01062004 A filha, tenho paciência não kkkkkkkkk COVID 762 pegaram eles 🤪
9	OrelhaM11	95	NaN	6	1	2	2021-05-07 23:59:29+00:00	Twitter for Android	Amanhã é diante guerra nas quartas 🔥 E pra deixar claro a única coisa que conseguiu parar o Borussia foi o Covid !! 100% Focado no jogo de amanhã e se Deus permitir, vamos para a semi.. 🔥🖤💛
10	JornalistaRuim	16	Banheiro	0	0	0	2021-05-07 23:59:26+00:00	Twitter for Android	@BlogdoNoblat Tá pouco deveria subir pra 100 e colocar como Covid
11	epocanegocios	231765	Brasil	3	3	0	2021-05-07 23:59:25+00:00	Twitter Web App	Fiocruz: pandemia de covid-19 faz vítimas cada vez mais jovens https://t.co/nZb0NLQCXs
12	COVID19BrUpdate	149	NaN	0	0	0	2021-05-07 23:59:25+00:00	COVID-19BrUpdate	Mais 4911 novo(s) caso(s) reportados no país. Agora com um total de 15087360 casos, sendo 1027489 casos ativos #covid19 #COVID_19 #covid19brasil #coronavirusnobrasil
13	alohamec7	1603	Rio de Janeiro, Brasil	0	0	0	2021-05-07 23:59:22+00:00	Twitter for iPhone	Essa porra desse COVID maldito 😭😭
14	moacircavalcant	87	NaN	0	0	0	2021-05-07 23:59:18+00:00	Twitter for Android	Verdade! O governo de PE DEVE dizer o que fez com BILHÕES que recebeu da UNIÃO. Para Pedro Eurico, CPI da Covid deve dar respostas à sociedade sobre gastos com cloroquina e atraso na vacinação https://t.co/CVvDo5gQdw
15	joaovitorjv	2846	NaN	11	1	0	2021-05-07 23:59:18+00:00	Twitter for Android	o desdém da mídia com o Cassiano mesmo na morte dele é revoltante. 15 dias atrás falaram que ele tava com pneumonia e na espera por uma vaga no CTI ou UTI. agora, não conseguem apurar se ele pegou Covid ou se foi a tal pneumonia dita antes..

Figura B.1: Visão geral das primeiras linhas do Dataframe de dados coletados sem tratamento.

C. ALGORITIMO PARA AUTOMATIZAÇÃO DA IDENTIFICAÇÃO DE INFORMAÇÕES DE MAIOR RELEVÂNCIA

Código fonte:

```
# Importação de bibliotecas para tabulação de dados
import numpy as np
import pandas as pd

# Importação de bibliotecas para normalizações, preparações e tokenizações de dados
import re
import unidecode
import gensim
from natsort import index_natsorted
from nltk.tokenize.treebank import TreebankWordDetokenizer

# Importação de bibliotecas para trabalhar com supervisão fraca
from snorkel.labeling import labeling_function
from sklearn.model_selection import train_test_split
from snorkel.labeling import labeling_function,PandasLFApplier,LFAnalysis

# Importação de bibliotecas para PLN no idioma Português
import spacy
nlp = spacy.load("pt_core_news_sm")

# Importação de bibliotecas para Visualização de dados
import plotly.express as px
import seaborn as sns
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

# Importação de bibliotecas para validar modelo através de Redes Neurais Recorrentes
import keras
from sklearn.model_selection import train_test_split
import tensorflow as tf
from keras.models import Sequential
from keras import layers
from keras.optimizers import RMSprop,Adam
from tensorflow.keras.optimizers import Adam, SGD, RMSprop
from keras.preprocessing.text import Tokenizer
from keras_preprocessing.sequence import pad_sequences
from keras import regularizers
from keras import backend as K
from keras.callbacks import ModelCheckpoint

```

```
# Importação de bibliotecas complementares para mensurar performance
from time import sleep, perf_counter
from threading import Thread

# Carregamento dos dados coletados em dataframe
df = pd.read_csv('./dataset-covid.csv', lineterminator='\n')

# Aplicação de normalizações iniciais
start_time = perf_counter()
df["Tweet"] = df["Tweet"].str.replace(r'https?://[^\s<>"]+|www\.[^\s<>"]+', "")
df['lowercase'] = df['Tweet'].apply(lambda x: "\
".join(unidecode.unidecode(x).lower() for \
x in x.split()))
end_time = perf_counter()
print(f'Tempo de {end_time- start_time: 0.2f} segundos paa completar tarefa.')

# Conferência de tipo de dados do dataframe
df.info()

# Coerção de tipos de dados
df = df.astype({"Followers":"int","No_Likes":"int","No_RT":"int","No_Replies":"int"})

# Criar Nuvem de palavras com todo dataset
def show_wordcloud(data, title=""):
    text = " ".join(t for t in data.dropna())
    stopwords = set(STOPWORDS)
    stopwords.update(['a', 'à', 'adeus', 'agora', 'aí', 'ainda', 'além', 'algo',\
                      'alguém', 'algum', 'alguma', 'algumas', 'alguns', 'ali', \
                      'ampla', 'amplas', 'amplo', 'amplos', 'ano', 'anos', \
                      'ante', 'antes', 'ao', 'aos', 'apenas', 'apoio', 'após', \
                      'aquela', 'aquelas', 'aquele', 'aqueles', 'aqui', \
                      'aquilo', 'área', 'as', 'às', 'assim', 'até', 'atrás', \
                      'através', 'baixo', 'bastante', 'bem', 'boa', 'boas', \
                      'bom', 'bons', 'breve', 'cá', 'cada', 'catorze', 'cedo', \
                      'cento', 'certamente', 'certeza', 'cima', 'cinco', \
                      "t", "co", "https", "amp", "U", "para", "até", "muito", \
                      "uma", "disse", "pois", "o", "não", "mais", "de", \
                      "sendo", "ele", "que", "tá", "e", "eu", "q", "dos", \
                      "deu", "da", "n", "nhttps", "drt", "tbm", "isso", "ela", \
                      "por", "tava", "esse", "essa", "este", "esta", "está", \
                      "ao", "vc","pq" "ou", "são", "vai", "agora", "tudo", \
                      "dos", "das", "do", "da", "já", "fazer", "ta", "ver", \
                      "ou", "é", "dia", "ser", "todo", "alguém", "era", "mas", \
                      "outro", "outra", "foi", "estão", "seria", "eles", "à", \
                      "há", "na", "os", "estou", "estar", "estava", "quando", \
                      "todos", "um", "os", "vão", "sua", "as", "né", "vez", \
                      "min", "se", "em", "sem", "um", "aqui", "ainda", "nos", \
                      "aqui", "antes", "como", "bem", "p", "você", "pq", "ai", \
                      "aí", "", "quem", "nem", "meu", "pra", "pelo", "faz",\
                       "pro", "nEu", "ir", "vcs", "desde", "conta", "la", \
                      "lá", "c", "nao", "nE", "tb", "nMas", "u200d", "nNão", \
```

```
                      "tb", "mt", "voltar", "ja", "realmente", "ia", "msm", \
                      "agr", "eh", "tô", "kkkkkk", "cu", "kkk", "covid", \
                      "NaN"])
    wordcloud = WordCloud(stopwords=stopwords, scale=6, max_font_size=40, \
                          max_words=1800,background_color="#ffffff", \
                          colormap='gist_yarg').generate(text)
    fig = plt.figure(1, figsize=(20,20))
    plt.axis('off')
    fig.suptitle(title, fontsize=20)
    fig.subplots_adjust(top=2.3)

    plt.imshow(wordcloud, interpolation='bilinear')
    plt.style.use('ggplot')
    plt.show()

show_wordcloud(df['Tweet'] , title = 'Prevalent words in collected tweets')

def plot_count(feature, title, df, size=1, ordered=True):
    f, ax = plt.subplots(1,1, figsize=(4*size,4),)
    total = float(len(df))
    if ordered:
        g = sns.countplot(df[feature], \
        order = df[feature].value_counts().index[:30], palette='RdBu')
    else:
        g = sns.countplot(df[feature], palette='RdBu')
    g.set_title("Number and percentage of {}\n".format(title))
    if(size > 2):
        plt.xticks(rotation=90, size=10)
    for p in ax.patches:
        height = p.get_height()
        ax.text(p.get_x()+p.get_width()/2. ,\
        height, ' {:1.2f}% '.format(100*height/total), ha="center")
    plt.show()

# Verificação dos usuários mais recorrentes
plot_count('Username' , "UserName" , df , 6)

# Verificação dos locais mais comuns de origem da publicação
plot_count("Location", "User location", df,6)

# Verificação do quantitativo de dispositivos de origem
plot_count("Source_device", "Source", df,6)

## Lista dos 15 Tweets com maior número de Retweets
df.sort_values(
   by="No_RT",
   key=lambda x: np.argsort(index_natsorted(df["No_RT"]))
).iloc[::-1].head(15)[["Username", "No_RT", "lowercase"]]

## Lista dos 25 Tweets com maior número de Réplicas
df.sort_values(
```

```
    by="No_Replies",
    key=lambda x: np.argsort(index_natsorted(df["No_Replies"]))
).iloc[::-1].head(25)[["Username", "No_Replies", "lowercase"]]

#########
#########

def depure_data(data):

    #Removing URLs with a regular expression
    url_pattern = re.compile(r'https?://\S+|www\.\S+')
    data = url_pattern.sub(r'', data)

    # Remove Emails
    data = re.sub('\S*@\S*\s?', '', data)

    # Remove new line characters
    data = re.sub('\s+', ' ', data)

    # Remove distracting single quotes
    data = re.sub("\'", "", data)

    # Remove dpunctuation signs
    data = re.sub("#", " ", data)

    return data

temp = []
# Splitting pd.Series to list
data_to_list = df_train["Tweet"].values.tolist() # raw collected tweeters text
for i in range(len(data_to_list)):
    temp.append(depure_data(data_to_list[i]))
list(temp[:30])

def sent_to_words(sentences):
    for sentence in sentences:
        yield(gensim.utils.simple_preprocess(str(sentence), deacc=True))


data_words = list(sent_to_words(temp))
print(data_words[:10],'\n')

# Destokenização
def detokenize(text):
    return TreebankWordDetokenizer().detokenize(text)

data = []
for i in range(len(data_words)):
    data.append(detokenize(data_words[i]))
print(data[:5])

```

```
# Coversão lista de textos em arrays
data = np.array(data)

def recover_label_name(x):
    if x == 0:
        return "HIGHVALUE"
    elif x == 1:
        return "LOWVALUE"
    else:
        return "ABSTAIN"

df_train['valuation'] = df_train['Labels'].apply(lambda x: recover_label_name(x))

df_train[['Tweet', 'valuation']].head() # to check

pos = df_train['valuation'].value_counts()
print(pos)

def plot_info(df, feature, title):
    counts = df[feature].value_counts()
    percent = counts/sum(counts)
    print(percent,"\n")
    fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(12, 5))

    counts.plot(kind='bar', ax=ax1, color='green')
    percent.plot(kind='bar', ax=ax2, color='blue')
    ax1.set_ylabel(f'Counts : {title} qty', size=12)
    ax2.set_ylabel(f'Percentage : {title} qty', size=12)
    plt.suptitle(f"Quality of Information: {title}\n")
    plt.tight_layout()
    plt.show()

plot_info(df_train, 'valuation', 'Collected Tweets')

# Definicão dos rótulos do modelo
labels = np.array(df_train["valuation"])
y = []
for i in range(len(labels)):
    if labels[i] == "ABSTAIN":
        y.append(0)
    if labels[i] == "LOWVALUE":
        y.append(1)
    if labels[i] == "HIGHVALUE":
        y.append(2)
y = np.array(y)
labels = tf.keras.utils.to_categorical(y, 3, dtype="float32")
del y

# Conversão de palavra em vetor
max_words = 5000
max_len = 200
```

```
245
246 tokenizer = Tokenizer(num_words=max_words)
247 tokenizer.fit_on_texts(data)
248 sequences = tokenizer.texts_to_sequences(data)
249 tweets = pad_sequences(sequences, maxlen=max_len)
250 print(tweets)
251
252 # Divisão dos dados
253 X_train, X_test, y_train, y_test = train_test_split(tweets,labels, random_state=0)
254 print (f"Our data split form:\n")
255 print(f"X_train: ",len(X_train))
256 print(f"X_test: ",len(X_test))
257 print(f"y_train: ",len(y_train))
258 print(f"y_test: ", len(y_test))
259
260 ###
261
262 model1 = Sequential()
263 model1.add(layers.Embedding(max_words, 20))
264 model1.add(layers.LSTM(15,dropout=0.5))
265 model1.add(layers.Dense(3,activation='softmax'))
266
267
268 model1.compile(optimizer='rmsprop',loss='categorical_crossentropy', \
269 metrics=['accuracy'])
270 #Implementing model checkpoins to save the best metric and do not
271 lose it on training.
272 checkpoint1 = ModelCheckpoint("best_model1.hdf5", monitor='val_accuracy', \
273 verbose=1,save_best_only=True, mode='auto', period=1,save_weights_only=False)
274 history1 = model1.fit(X_train, y_train, epochs=10,\
275 validation_data=(X_test, y_test),callbacks=[checkpoint1])
276
277 # summarize history for accuracy
278 plt.plot(history1.history['accuracy'])
279 plt.plot(history1.history['val_accuracy'])
280 plt.title('LSTM model accuracy')
281 plt.ylabel('accuracy')
282 plt.xlabel('epoch')
283 plt.legend(['train', 'test'], loc='upper left')
284 plt.show()
285
286 # summarize history for accuracy
287 plt.plot(history1.history['loss'])
288 plt.plot(history1.history['val_loss'])
289 plt.title(' LSTM model loss')
290 plt.ylabel('loss')
291 plt.xlabel('epoch')
292 plt.legend(['train', 'test'], loc='upper right')
293 plt.show()
294
295 import keras
```

```
best_model = keras.models.load_model("./best_model1.hdf5")

lstmpredict = model1.predict(X_test)
print(lstmpredict)

test_loss, test_acc = best_model.evaluate(X_test, y_test, verbose=2)
print('Model accuracy: ',test_acc)

test_loss, test_acc = best_model.evaluate(lstmpredict, y_test, verbose=2)
print('Model accuracy: ',test_acc)

# Plot the classification_report
from sklearn.metrics import classification_report
print(classification_report(np.argmax(y_test, axis=1),np.argmax(lstmpredict, \
axis=1)))

model = Sequential()
model.add(layers.Embedding(max_words, 40, input_length=max_len))
model.add(layers.Bidirectional(layers.LSTM(20,dropout=0.6)))
model.add(layers.Dense(3,activation='softmax'))

# Compile the model
model.compile(optimizer='rmsprop',loss='categorical_crossentropy', \
metrics=['accuracy'])
# Save the Modek
checkpoint2 = ModelCheckpoint("best_model2.hdf5", monitor='val_accuracy', \
verbose=1,save_best_only=True, mode='auto', period=1,save_weights_only=False)
# Train the Model
history = model.fit(X_train, y_train, epochs=10,validation_data=(X_test, y_test),\
callbacks=[checkpoint2])

# summarize history for accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('BiLSTM model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

# summarize history for accuracy
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('BiLSTM model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper right')
plt.show()

best_model = keras.models.load_model("./best_model2.hdf5")

```

```
predictions = best_model.predict(X_test)
print(predictions)

test_loss, test_acc = best_model.evaluate(X_test, y_test, verbose=2)
print('Model accuracy: ',test_acc)

# Plot the classification_report
from sklearn.metrics import classification_report
print(classification_report(np.argmax(y_test, axis=1),np.argmax(predictions, \
axis=1)))

########################
#### Aplicação do modelo ####

# Prediction labels
info = ['abstain','lowvalue','highvalue']

tweet_text_from_dataset = 'Covid: Média móvel de mortes mantém tendência de \
queda há quatro dias no Brasil'

sequence = tokenizer.texts_to_sequences([tweet_text_from_dataset])
test = pad_sequences(sequence, maxlen=max_len)
info[np.around(best_model.predict(test), decimals=0).argmax(axis=1)[0]]


tweet_text_unknown = 'Olha, sinceramente eu não sei como o covid ainda não \
comeu o c* dessa família, tem umas dez máscaras descartáveis que foram lavadas \
na cerca!\nJá falei umas mil vezes, já dei pff2 pra geral (ninguém usou), \
que difícil viu...'

sequence = tokenizer.texts_to_sequences([tweet_text_unknown])
test = pad_sequences(sequence, maxlen=max_len)
info[np.around(best_model.predict(test), decimals=0).argmax(axis=1)[0]]

```