Universidade de Brasília

Faculdade de Tecnologia

Uma ferramenta baseada em Aprendizado de Máquina para Detecção de Notícias Falsas

Lucas Aquino Costa

PROJETO FINAL DE CURSO

ENGENHARIA DE COMPUTAÇÃO

Brasília

2024

Universidade de Brasília
Faculdade de Tecnologia

Uma ferramenta baseada em Aprendizado de Máquina para Detecção de Notícias Falsas

Lucas Aquino Costa

Projeto Final de Curso submetido como requisito parcial para obtenção do grau de Engenheiro de Computação.

Orientador: Prof. Dr. Daniel Guerreiro e Silva

Brasília
2024

FICHA CATALOGRÁFICA

Aquino Costa, Lucas.

Uma ferramenta baseada em Aprendizado de Máquina para Detecção de Notícias Falsas / Lucas Aquino Costa; orientador Daniel Guerreiro e Silva. -- Brasília, 2024.

63 p.

Projeto Final de Curso (Engenharia de Computação) -- Universidade de Brasília, 2024.

1. Máquinas de Vetor Suporte. 2. Extensão Google Chrome. 3. Modelo. 4. Notícias Falsas. 5. Conjunto de dados.I. Guerreiro e Silva, Daniel, orient. II. Título.

Universidade de Brasília
Faculdade de Tecnologia

Uma ferramenta baseada em Aprendizado de Máquina para Detecção de Notícias Falsas

Lucas Aquino Costa

Projeto Final de Curso submetido como requisito parcial para obtenção do grau de Engenheiro de Computação.

Trabalho aprovado. Brasília, 18 de setembro de 2024:

Prof. Dr. Daniel Guerreiro e Silva,
UnB/FT/ENE
Orientador

Prof. Dr. Fábio Lúcio Lopes de Mendonça,
UnB/FT/ENE
Examinador interno

Prof. Dr. José Edil Guimarães de Medeiros,
UnB/FT/ENE
Examinador interno

Agradecimentos

Aos meus irmãos, pelo suporte diário na minha vida; ao meu tio e ao meu avô, pelo incentivo e investimento na minha carreira acadêmica; e, em especial, aos meus pais, que são o motivo de eu querer entrar na UnB. À minha namorada, por ser um porto seguro ao longo dessa jornada, e à sua família, por me receber como um filho. Agradeço ao professor Daniel Guerreiro por me guiar até aqui. Por fim, sou grato a todos os meus amigos que compartilharam minhas vitórias e minhas angústias ao longo de todos esses anos.

Resumo

As notícias falsas (*fake news*) constituem um tipo de conteúdo informativo que se baseia em informações inverídicas ou enganosas, apresentadas como se fossem verdadeiras. Essas informações podem ser completamente fabricadas, sem qualquer relação com a realidade, ou podem ser distorções de fatos, frequentemente disseminadas com o objetivo de influenciar opiniões, manipular emoções ou obter vantagens políticas. Essas notícias podem se manifestar por meio de textos, imagens, vídeos ou outras formas de conteúdo informativo, neste projeto, o foco estará nas notícias falsas apresentadas na forma de texto. Há muito tempo, esforços têm sido empreendidos para combater a proliferação dessas notícias falsas. Este projeto tem como objetivo apresentar uma solução que possa ser distribuída para consumidores de notícias. Para tanto, foi utilizado um trabalho anterior como base, com o aprimoramento e adição de novas funcionalidades, visando disponibilizar essa ferramenta ao público em geral. Foram empregadas soluções baseadas em Máquinas de Vetores de Suporte, o desenvolvimento de uma extensão de navegador construído com JavaScript, e uma API implementada em Python utilizando Flask. Ao final, é realizada uma análise detalhada do uso da extensão em diferentes notícias, comprovando que é possível disponibilizar essa ferramenta para o usuário final. Ademais, testes realizados com duas bases de dados indicaram o conjunto *FakeRecogna* com os melhores resultados.

Palavras-chave: Máquinas de Vetor Suporte. Extensão Google Chrome. Modelo. Notícias Falsas. Conjunto de dados.

Abstract

Fake news constitutes a type of informational content based on false or misleading information presented as if it were true. This information can be completely fabricated, with no relation to reality, or may be distortions of facts, often disseminated with the intention of influencing opinions, manipulating emotions, or gaining political advantages. These news pieces can manifest in the form of text, images, videos, or other types of informational content; in this project, the focus will be on fake news presented in text form. Efforts have long been made to combat the spread of fake news. This project aims to present a solution that can be distributed to news consumers. For this purpose, a previous work was used as a foundation, with improvements and the addition of new functionalities to make this tool available to the general public. Solutions based on Support Vector Machines were employed, along with the development of a browser extension built with JavaScript, and an API implemented in Python using Flask. Finally, a detailed analysis of the extension's use on different news articles is conducted, proving that it is possible to provide this tool to the end user. Furthermore, tests conducted with two datasets indicated that the *FakeRecogna* set yielded the best results.

Keywords: Support Vector Machines. Google Chrome extension. Model. Fake News. Dataset.

Lista de figuras

Lista de tabelas

Sumário

1 Introdução

Com o avanço das redes sociais na última década, podemos observar o aumento das notícias falsas. Elas são informações fabricadas ou distorcidas e apresentadas como verdadeiras. Elas podem ser divulgadas intencionalmente para enganar ou manipular o público, ou de maneira não intencional, devido à falta de verificação dos fatos (Fletcher *et al.*, 2018).

Sites e redes sociais são grandes agentes na disseminação de notícias falsas, as quais podem proliferar rapidamente. "A falsidade se disseminou significativamente mais longe, mais rápido, mais profundamente e mais amplamente do que a verdade em todas as categorias de informação" (Vosoughi; Roy; Aral, 2018). Isso se dá pelo fato de as notícias falsas serem mais novas e apelarem ao desejo humano de compartilhar informações inéditas (Vosoughi; Roy; Aral, 2018). Além disso, de acordo com (Lai, 2022), "A sobrecarga de informações nas redes sociais cria um ambiente caótico e avassalador, tornando difícil para as pessoas distinguirem o fato da ficção, dessa forma cada dia que passa as pessoas estão cada vez mais divulgando e consumindo esse tipo de informação".

Na política, as notícias falsas têm um impacto significativo ao distorcer percepções e influenciar decisões. Informações falsas podem ser disseminadas de forma estratégica para promover ou prejudicar a reputação de um político ou partido. Vimos, no ano de 2018, que notícias foram espalhadas no WhatsApp sobre os dois candidatos à presidência do segundo turno, com alegações falsas sobre Fernando Haddad, candidato do partido de esquerda, sendo retratado como comunista e associado a regimes autoritários como Cuba e Venezuela. Da mesma forma, divulgou-se que Jair Bolsonaro, candidato do partido de direita, poderia ter encenado um ataque contra si mesmo durante a campanha (Scarabelli, 2018).

Além disso, essas notícias também estão presentes na saúde, afetando comportamentos e a implementação de intervenções baseadas em evidências. De acordo com a OMS, a desinformação pode agravar situações de crise de saúde pública, distorcendo a percepção pública sobre a eficácia e segurança das vacinas, o que aumenta a hesitação vacinal. Segundo o artigo "Social Media and the Influence of Fake News on Global Health Interventions: Implications for a Study on Dengue in Brazil" (Gupta *et al.*, 2023), uma equipe de pesquisa responsável pela redução da incidência de dengue em Fortaleza enfrentou resistência da comunidade devido à circulação de notícias falsas nas redes sociais. Isso evidencia como a desinformação cria obstáculos significativos para a saúde pública. Ademais, conforme (Union for International Cancer Control, 2021), a disseminação de informações médicas falsas ou enganosas pode ter consequências diretas na vida das pessoas, induzindo-as a acreditar em curas não comprovadas para o câncer, o que pode levar a decisões prejudiciais à saúde.

Dessa forma, é necessario definir nesse trabalho o que é uma noticia falsa, para que dessa forma essa definição seja usada como base para poder analisar o resultados finais do uso da aplicação. Sendo assim, a sua definição vai ser baseada no conjunto de dados do *Corpus FAKE.BR* (Monteiro *et al.*, 2018). Onde um dos sinais mais comuns de notícias falsas inclui a ausência de autoria, o que impede a identificação de uma fonte confiável. Além disso, esses textos costumam apresentar erros gramaticais frequentes, o que reflete falta de cuidado na produção. Outro aspecto marcante é o uso de títulos sensacionalistas, que buscam atrair atenção de forma exagerada. As notícias também costumam trazer afirmações muito fortes e impactantes, com o objetivo de provocar reações imediatas. Por fim, observa-se o uso excessivo de pontos de exclamação e letras maiúsculas, estratégias visuais que visam destacar o conteúdo de maneira forçada e apelativa.

Sabendo disso, é necessária a intervenção para o combate a essa prática, e entre as inúmeras estratégias, pode-se empregar técnicas de Inteligência Artificial para desenvolver ferramentas de alerta ao público geral. Dessa forma, com base no trabalho "Estudo e Aplicação de SVMs na Detecção de Fake-News" (Silva; Fagundes, 2021) propõe-se uma ferramenta de detecção de notícias falsas para navegadores *Web*. Foi configurado o modelo sugerido no estudo, o qual foi retreinado utilizando um novo conjunto de dados e outro já utilizado no trabalho: o *FakeBR* (Monteiro *et al.*, 2018) e o *FakeRecogna* (Garcia; Afonso; Papa, 2022). Os modelos gerados foram armazenados e integrados a uma Interface de Programação de Aplicações (API), permitindo seu consumo por uma extensão de navegação do Google Chrome. Com isso, uma ferramenta que antes estava indisponível ao público geral agora pode ter fácil acesso, possibilitando a democratização de uma tecnologia importante para o combate à desinformação.

1.1 Motivação

A motivação para este trabalho surge do interesse em auxiliar, cada vez mais, as pessoas a combaterem as notícias falsas, tendo em vista que sua propagação tem aumentado nos dias atuais, especialmente com o avanço da inteligência artificial e as técnicas *deep-fake*. Este trabalho foi baseado no estudo realizado pelos alunos Daniela Prass e Rodrigo Andrade (Silva; Fagundes, 2021), que testa o uso de Máquina de Vetores de Suporte na detecção de notícias falsas. No entanto, tal estudo só se ocupou de validar a referida técnica como pontencial solução. Para avaliar o combate à desinformação, é necessário disponibilizar essa técnica como uma ferramenta ao público em geral.

Um trabalho relacionado seria o proposto em (Santos *et al.*, 2023), onde os autores utilizaram os dados de (Garcia; Afonso; Papa, 2022) como base para o *ChatGPT*. Nesse trabalho, o *ChatGPT*, um assistente geral de converção baseado em Inteligência Artificial, foi responsável por determinar se uma notícia é falsa ou não. No trabalho atual, a ideia é

disponibilizar e treinar diretamente um modelo com as bases de dados derivadas de (Garcia; Afonso; Papa, 2022) e (Monteiro *et al.*, 2018).

1.2 Objetivo

O presente trabalho tem como objetivo desenvolver uma ferramenta de acesso ao público em geral que indica a chance de uma notícia ser falsa, de maneira automática. Para isso, inicia-se com a atualização do banco de dados de um estudo anterior (Silva; Fagundes, 2021), utilizando tanto o conjunto de dados original quanto um novo conjunto. O modelo resultante será treinado, armazenado e disponibilizado por meio de uma API. Em seguida, será desenvolvida uma extensão para o navegador Google Chrome, que consumirá as informações fornecidas pela API. Para validar a operação da extensão, serão conduzidos testes em sites de notícias selecionados aleatoriamente, cujos resultados servirão para comparar os bancos de dados, incluindo o *FakeBR* e o novo conjunto de dados *FakeRecogna*. Adicionalmente, será analisado o desempenho e a usabilidade da extensão.

1.3 Organização

O restante deste trabalho está organizado da seguinte forma: no segundo capítulo, é apresentada a fundamentação teórica que sustenta os experimentos e os resultados obtidos; no terceiro capítulo, descreve-se o processo de retreinamento do modelo indicado em (Silva; Fagundes, 2021), a construção da API, o desenvolvimento da extensão para o navegador Google Chrome, e os testes realizados com diferentes bancos de dados em diversos sites de notícias; o quarto capítulo apresenta os resultados obtidos; o quinto capítulo discute as conclusões do estudo e sugere direções para trabalhos futuros.

2 Fundamentação Teórica

Este capítulo tem como objetivo apresentar as razões por trás da escolha do tema, além de explorar as ideias e conceitos fundamentais para a implementação do classificador de notícias falsas e da extensão desenvolvida. Ao final deste capítulo, espera-se que o leitor possua as bases teóricas necessárias para compreender a aplicação prática que será discutida no capítulo 3.

2.1 Detecção Automática de Notícias Falsas

Uma das maiores dificuldades para um consumidor de notícias é detectar se ela é falsa ou não. Com o avanço das ferramentas generativas e de edição baseadas em inteligência artificial, essa tarefa se torna cada vez mais desafiadora, exigindo uma atenção minuciosa aos detalhes que podem determinar a veracidade de uma notícia. De acordo com o (Musgrove *et al.*, 2018), podemos analisar a notícia da seguinte maneira antes de determinar sua veracidade:

1. Verificar a credibilidade e confiabilidade da fonte da notícia;
2. Verificar as credenciais do autor e o histórico do autor;
3. Buscar a notícia em outras mídias confiáveis para verificar se todos os veículos estão reportando o ocorrido de maneira consistente.

Grandes plataformas associadas à empresa Meta (Meta, 2024) têm desenvolvido mecanismos para verificar e remover notícias falsas em suas redes. Um exemplo é o Instagram, que utiliza técnicas de *machine learning* para identificar postagens falsas, aplicando um selo informativo sobre o conteúdo (Instagram, 2022).

Além disso, estudos como o de (Sharma; Saran; Patil, 2020) abordam essa problemática ao classificar notícias online como verdadeiras ou falsas utilizando Inteligência Artificial, Processamento de Linguagem Natural e Aprendizado de Máquina, além de verificar a autenticidade dos sites que publicam essas notícias. O estudo de (Abdulrahman; Baykara, 2020) demonstrou que é possível classificar notícias falsas por meio de uma rede neural convolucional, alcançando uma precisão entre 81% e 100%. Existem também trabalhos que se distanciam da classificação tradicional, predominantemente aplicada na área política, buscando desenvolver classificadores mais genéricos e de uso geral, utilizando abordagens de *ensemble* de aprendizado de máquina em diferentes contextos (Ahmad *et al.*, 2020). No Brasil, destaca-se o estudo de (Monteiro *et al.*, 2018), que apresenta um novo conjunto de dados público e real de notícias verdadeiras e falsas rotuladas em português, além de realizar

uma análise abrangente dos métodos de aprendizado de máquina aplicados à detecção de notícias falsas.

2.2 Aprendizado de Máquina (ML)

O principal objetivo desta seção é apresentar conceitos básicos sobre Aprendizado de Máquina, baseados em (Goodfellow; Bengio; Courville, 2016).

2.2.1 Algoritmo

Um algoritmo de aprendizado de máquina é um algoritmo capaz de aprender a partir de dados. (Mitchell, 1997), amplamente reconhecido como uma das principais autoridades no campo do aprendizado de máquina, fornece uma definição concisa sobre o que é aprender: “Diz-se que um programa de computador aprende com a experiência E em relação a uma classe de tarefas T e medida de desempenho P, se seu desempenho nas tarefas em T, conforme medido por P, melhora com a experiência E”. Pode-se imaginar uma grande variedade de experiências E, tarefas T e medidas de desempenho P, e não vai ser definido formalmente o que pode ser usado para cada uma dessas entidades, mas será fornecido descrições intuitivas e exemplos dos diferentes tipos de tarefas, medidas de desempenho e experiências que podem ser usados para construir algoritmos de aprendizado de máquina.

O aprendizado de máquina nos permite enfrentar tarefas que são complexas demais para serem resolvidas com programas fixos, escritos e projetados por humanos. Na definição formal de "tarefa" (T), o aprendizado em si não é a tarefa, mas o meio para alcançar a habilidade desejada. Por exemplo, ensinar um robô a andar não é a tarefa; andar é a tarefa, e o aprendizado é o meio para alcançar essa habilidade.

Muitas tarefas podem ser resovidas com aprendizado de máquina, algumas delas podem ser:

- Classificação, uma tarefa em que um programa de computador determina a qual categoria uma entrada pertence;
- Regressão, uma tarefa de aprendizado de máquina onde o programa prevê um valor numérico com base em entradas;
- Tradução automática, também conhecida como *Machine translation*, é uma tarefa em que o programa converte uma sequência de símbolos de uma língua para outra, como traduzir do inglês para o francês.

Diversas outras tarefas e tipos de tarefas são possíveis no aprendizado de máquina. Os exemplos apresentados servem para ilustrar as capacidades dessa área, e não para definir uma lista exaustiva ou obrigatória de tarefas específicas.

Para avaliar as capacidades de um algoritmo de aprendizado de máquina, devemos definir uma medida quantitativa de seu desempenho. Normalmente, essa medida de desempenho P é específica para a tarefa T executada pelo sistema. Em tarefas como classificação e transcrição, a acurácia mede a proporção de exemplos corretamente classificados, enquanto a taxa de erro avalia as classificações incorretas. Para problemas como estimação de densidade, onde acurácia não se aplica, são usadas métricas de desempenho contínuas, como a média do logaritmo das probabilidades atribuídas aos exemplos pelo modelo. Na maioria dos casos estamos interessados em como o algoritmo de aprendizado de máquina se comporta em dados que ele nunca viu antes, pois isso determina seu desempenho no mundo real. Portanto, avaliamos essas medidas de desempenho usando um conjunto de teste, que é um conjunto de dados separado dos dados usados para o treinamento do sistema de aprendizado de máquina. A escolha da medida de desempenho pode parecer simples e objetiva, mas frequentemente é difícil selecionar uma que corresponda bem ao comportamento desejado do sistema. Por exemplo, ao realizar uma tarefa de transcrição, devemos medir a acurácia na transcrição de sequências inteiras ou usar uma medida de desempenho mais detalhada que dê crédito parcial por transcrever alguns elementos corretamente? Em uma tarefa de regressão, devemos penalizar o sistema mais por erros médios frequentes ou por erros muito grandes, porém raros? Cada escolha depende da aplicação.

2.2.2 Capacidade, Sobreajuste e Subajuste

O principal desafio do aprendizado de máquina é garantir que o algoritmo generalize bem para novas entradas, não apenas para aquelas usadas no treinamento. A generalização é medida pelo erro de teste, que reflete o desempenho do modelo em dados não vistos, sendo estimado através de um conjunto de teste separado do conjunto de treinamento. A diferença fundamental entre otimização e aprendizado de máquina é que, além de minimizar o erro de treinamento, queremos minimizar o erro de generalização para garantir um bom desempenho no mundo real.

Os dados de treinamento e teste são gerados a partir de uma distribuição de probabilidade comum chamada processo gerador de dados. Supondo que os exemplos são independentes e distribuídos de forma idêntica (i.i.d.), podemos estudar a relação entre erro de treinamento e erro de teste usando essa distribuição compartilhada, facilitando a análise matemática do desempenho do modelo.

O erro de treinamento esperado de um modelo selecionado aleatoriamente é igual ao erro de teste esperado, pois ambos são calculados a partir do mesmo processo de amostragem da distribuição de probabilidade dos dados. Ao amostrar repetidamente de uma distribuição $p(x, y)$ para gerar os conjuntos de treinamento e teste, o erro esperado para um modelo com um valor fixo w será o mesmo em ambos os conjuntos. A única diferença entre as condições é o nome atribuído aos conjuntos amostrados, reforçando que a expectativa é formada a

partir do mesmo processo estatístico.

É claro que, quando usamos um algoritmo de aprendizado de máquina, não fixamos os parâmetros antecipadamente para depois amostrar ambos os conjuntos de dados. Primeiro, amostramos o conjunto de treinamento, usamos esse conjunto para escolher os parâmetros com o objetivo de reduzir o erro no treinamento, e só depois amostramos o conjunto de teste. Nesse processo, o erro de teste esperado é maior ou igual ao valor esperado do erro de treinamento. Os fatores que determinam o desempenho de um algoritmo de aprendizado de máquina são sua capacidade de:

- Reduzir o erro de treinamento;
- Reduzir a diferença entre o erro de treinamento e o erro de teste.

Esses dois fatores correspondem aos dois principais desafios do aprendizado de máquina: subajuste (*underfitting*) e sobreajuste (*overfitting*). O subajuste ocorre quando o modelo não consegue obter um erro suficientemente baixo no conjunto de treinamento. O sobreajuste acontece quando a diferença entre o erro de treinamento e o erro de teste é muito grande.

2.2.3 Aprendizado Supervisionado

Essa subseção apresenta conceitos retirados de (Alpaydin, 2020).

Algoritmos de aprendizado supervisionado utilizam conjuntos de dados com características e rótulos associados para aprender. Por exemplo, o conjunto de dados "Iris"contém medições de plantas de íris e suas espécies. Um algoritmo supervisionado pode estudar essas medições e aprender a classificar as plantas em três espécies distintas com base nos dados fornecidos (Goodfellow; Bengio; Courville, 2016). Todo modelo supervisionado aborda um assunto específico obtido a partir de uma coleta de dados. Inicialmente, deve-se realizar um pré-processamento, no qual é feita uma análise preliminar dos dados para compreensão do problema. Em seguida, os dados são tratados de modo a se obter um conjunto no formato adequado para entrada no modelo. Por fim, esse conjunto de dados é dividido em três subconjuntos: treino, validação e teste.

Na fase subsequente, todos os modelos candidatos são treinados utilizando o conjunto de treinamento. Como resultado, obtém-se um conjunto de modelos treinados, capazes de responder à questão inicial. O desafio, então, consiste em selecionar o melhor modelo entre os candidatos. Para isso, é essencial escolher uma métrica de qualidade adequada. Todos os modelos candidatos realizam predições sobre o conjunto de validação e são avaliados segundo essa métrica, sendo ranqueados de acordo com suas pontuações. O modelo que obtiver a maior pontuação é designado como o melhor modelo entre os candidatos. Esse processo de avaliação é conhecido como validação cruzada. O melhor modelo é então treinado

novamente, utilizando também o conjunto de validação. Por fim, a qualidade do modelo final é medida, utilizando a métrica previamente escolhida, aplicada ao conjunto de teste.

O aprendizado supervisionado pode ser categorizado em dois tipos principais de problemas: classificação e regressão. A distinção entre eles reside no tipo de previsão que se busca obter. Quando o objetivo é agrupar as entradas em categorias específicas, utiliza-se um modelo supervisionado de classificação. Em contrapartida, se a intenção é prever um valor numérico a partir de uma determinada entrada, emprega-se um modelo supervisionado de regressão. Assim, a saída de um modelo de classificação é um valor discreto, enquanto a saída de um modelo de regressão é um valor contínuo. No contexto deste projeto, onde o objetivo é determinar se uma notícia é verdadeira ou falsa, configura-se um problema de classificação.

2.2.4 Máquina de Vetor de Suporte (SVM)

A *Support Vector Machine* (SVM) foi desenvolvida a partir do princípio da minimização de risco estrutural e baseia-se em um discriminante, ou seja, assume um modelo diretamente para o discriminante em vez de calcular verossimilhança, probabilidades a posteriori e densidades, como ocorre em outras abordagens (Vapnik, 1995). Esse método ganhou popularidade em relação aos baseados em verossimilhança, pois acredita-se que é mais fácil estimar o discriminante diretamente do que as densidades de probabilidade, como é feito em outras abordagens. Vapnik (1995) apoia essa ideia e argumenta que não faz sentido resolver um problema criando outro ainda mais complexo que o inicial.

O viés indutivo desse tipo de modelagem reside na suposição sobre a forma da fronteira que separa as classes (Alpaydin, 2020). Em outras palavras, o modelo SVM assume que existe uma fronteira linear capaz de separar os dados em diferentes classes. No contexto das SVMs, aprender significa ajustar os parâmetros de modo a maximizar a qualidade da estimativa da fronteira que distingue as regiões de cada classe. Essa fronteira é representada por um hiperplano, que é um subplano com dimensão m, menor que a dimensão original $m + 1$ (Alpaydin, 2020). Uma dimensão refere-se ao número mínimo de coordenadas necessárias para definir um ponto; assim, um hiperplano em um espaço bidimensional é uma linha (1 dimensão), enquanto em um espaço tridimensional é um plano (2 dimensões), e assim por diante. O hiperplano é determinado por um vetor de pesos aplicado a um subconjunto dos dados de treino, que forma os vetores de suporte.

Na classificação, os vetores de suporte são as instâncias localizadas próximas à fronteira que separa as classes, sendo essenciais para definir o hiperplano. A análise desses vetores permite estimar o erro de generalização, pois eles se situam na região crítica entre as duas classes, onde amostras podem estar classificadas incorretamente. As outras instâncias, que estão fora dessa zona de incerteza, tendem a ser classificadas corretamente e, portanto, não influenciam significativamente na definição da fronteira (Alpaydin, 2020).

O objetivo nas SVMs é não apenas que as instâncias estejam corretamente posicionadas em relação ao hiperplano, mas também que estejam a uma certa distância dele, chamada de margem. A maximização dessa margem é crucial para uma melhor generalização do modelo, pois quanto maior a distância das instâncias mais próximas ao hiperplano, menor é a probabilidade de novos dados serem classificados de forma errônea (Alpaydin, 2020).

Joachims (Joachims, 1998) foi um dos primeiros a aplicar SVMs na classificação de textos e destacou as vantagens dessa abordagem. Textos possuem um espaço dimensional elevado, com muitas features relevantes, o que pode tornar o processamento computacionalmente custoso, dependendo da técnica utilizada. Como as SVMs não são sensíveis à dimensionalidade dos dados de entrada, elas se mostram uma excelente escolha para lidar com esse tipo de problema.

2.3 Processamento de Linguagem Natural (NLP)

Linguagem é definida como um meio sistemático de comunicar ideias ou sentimentos por meio de signos convencionais, sejam eles sonoros, gráficos ou gestuais. Uma linguagem pode surgir naturalmente, como os idiomas, ou ser criada, como as linguagens de programação. Para ser considerada uma linguagem, é necessário que um conjunto de regras seja conhecido e utilizado por um grupo de pessoas para enviar e entender mensagens (Languages, 2021).

A linguagem natural, uma das características mais complexas e importantes dos seres humanos, pode ser rastreada até mais de 50 mil anos atrás e desempenhou um papel crucial na evolução da espécie, permitindo cooperação e comunicação entre indivíduos. Hoje, existem mais de sete mil idiomas falados no mundo, e essa diversidade representa um dos maiores desafios para a comunicação de máquinas inteligentes (Harari, 2015).

Processamento de Linguagem Natural (NLP) é um campo da inteligência artificial que busca traduzir a linguagem humana para a linguagem de máquina e vice-versa. Esse campo estuda tanto as capacidades cognitivas humanas quanto as estruturas linguísticas que compõem a linguagem. Para ensinar computadores a compreender e produzir linguagem, é necessário treiná-los utilizando regras linguísticas formais, uma vez que eles não podem aprender por assimilação natural (Russell; Norvig, 2010).

A tradução de linguagem por máquinas envolve técnicas de aprendizado de máquina para que possam reconhecer e entender textos. Esse processo é hierárquico, quebrando a informação em unidades menores para uma compreensão completa. Métodos como tokenização, remoção de *stopwords* e *stemming* são utilizados para lidar com as complexidades da linguagem natural, conforme a metodologia de Manning (Manning; Schütze, 1999).

2.3.1 Tokenização

Quando alguém deseja aprender uma nova língua, inicia pelo aprendizado do significado de cada palavra de forma individual. Posteriormente, é necessário compreender o significado das palavras dentro do contexto do texto, fornecendo assim um contexto mais rico para a informação. Portanto, é fundamental entender cada bloco do texto para poder apreender o contexto como um todo.

Baseando-se na obra de Manning (Manning; Schütze, 1999), podemos afirmar que, no contexto de NLP, o conjunto de textos analisados é denominado *corpus*. Quando o computador lê um texto do *corpus*, realiza um procedimento semelhante à leitura humana. Na primeira instância, o texto é separado em blocos de informação. Esses blocos, denominados *tokens*, são extraídos do texto original e processados. Essa técnica é chamada de Tokenização.

Na escrita tradicional, a ideia de separar blocos de informação é realizada por meio de pontuações e parágrafos. Entretanto, no caso da tokenização, é possível ir muito além, separando o texto por qualquer conjunto de símbolos previamente definido. Para esse processo, é crucial o conhecimento da linguagem utilizada, pois cada idioma possui uma maneira distinta de atribuir significado a uma palavra e de construí-la. Dessa forma, podem surgir problemas com palavras homógrafas, como a palavra "vela" (de barco) e "vela" (de cera), possuem significados diferentes mas o computador pode atribuir o mesmo significado à palavra. Além disso, metáforas podem alterar completamente a compreensão de um texto, já que o significado original é utilizado em outro sentido.

Após a escolha dos *tokens*, eles são utilizados para formar o vocabulário do *corpus*. Esse vocabulário é composto pelo conjunto de *tokens* únicos e serve como base para o modelo identificar termos. Construído na fase de treinamento, o vocabulário funciona como a memória do modelo, representando a linguagem aprendida pelo computador. Quando um novo texto é inserido no modelo treinado, ele é primeiro dividido em *tokens*, que são então comparados com o vocabulário. Se houver correspondência, o modelo entende o *token* e pode utilizá-lo em suas previsões. Caso contrário, diferentes regras são aplicadas aos *tokens* desconhecidos, como descartá-los ou identificá-los como termos desconhecidos, dependendo do problema específico.

Sendo assim, o principal desafio da tokenização reside na escolha dos *tokens* a serem utilizados. Intuitivamente, pode-se pensar que seria ideal utilizar todos os *tokens* do conjunto de treino, mas essa abordagem não é viável devido ao alto custo computacional associado a um *corpus* grande, que exige significativa capacidade de processamento para treinar o modelo e fazer inferências. Além disso, utilizar todos os *tokens* pode causar sobreajuste no modelo, que ocorre quando o modelo se ajusta tão bem aos dados de treinamento que perde a capacidade de generalizar para novos dados, resultando em um desempenho ruim em situações não vistas anteriormente. Para resolver esse problema, diversas técnicas de normalização de dados foram desenvolvidas.

2.3.2 Normalização

As *stopwords* são palavras frequentes em um idioma que não necessariamente agregam valor significativo ao sentido que um texto deseja transmitir. É altamente recomendável removê-las para reduzir o processamento do modelo, especialmente na fase de treinamento. Dessa maneira, o vocabulário retém palavras mais relevantes para sua compreensão.

Em sua maioria, as palavras consideradas *stopwords* são pronomes, preposições, conjunções e artigos. No entanto, não existe uma convenção fixa sobre quais palavras devem ser removidas, pois cada problema terá um conjunto específico definido. Uma estratégia comum é identificar as palavras com maior frequência e removê-las manualmente, avaliando se essa remoção prejudicará o sentido do texto. Essa avaliação deve ser realizada de forma minuciosa, pois a remoção indevida de qualquer palavra pode comprometer a qualidade do modelo. Além disso, um modelo que retorna um texto como resposta pode gerar resultados desconexos para o leitor final se a remoção não for feita adequadamente. Temos como alguns *stopwords*:

- **Preposições:** de, em, com, por, para, sem, sobre;
- **Conjunções:** e, ou, mas, porém, embora;
- **Pronomes:** ele, ela, isso, aquilo, que, qual;
- **Artigos:** o, a, os, as, um, uma, uns, umas.

2.4 Extensão do Navegador Google Chrome

Os conceitos apresentados nessa seção se fundamentam em (Mehta, 2016)

As Extensões do Google Chrome são complementos para o navegador *web* Google Chrome que auxiliam os usuários a realizarem diversas tarefas enquanto navegam pela *web*. Desde 2010, é possível criar e utilizar essas extensões. Para indivíduos com conhecimento prévio em programação *web*, a curva de aprendizado é significativamente reduzida, uma vez que essas extensões não introduzem novas ferramentas, mas utilizam conceitos já familiares. Além disso, embora extensões possam ser encontradas em outros navegadores, este trabalho focará exclusivamente nas extensões do Google Chrome, pois, de acordo com (Olhar Digital, 2024), ele é o navegador mais utilizado mundialmente, com 75,27% de participação de mercado.

Um ponto importante a ser observado é que as extensões de navegador são diferentes dos plugins de navegador. Enquanto as extensões de navegador são executadas dentro do contexto de segurança do navegador *host* (*software*), os plugins não são. Além disso, as extensões adicionam novas funcionalidades aos navegadores ao combinar recursos existentes já disponíveis nos navegadores (no caso das Extensões do Chrome, isso é feito usando a API é uma interface que permite a comunicação e integração entre diferentes softwares,

facilitando o uso de funcionalidades ou dados de outros sistemas de maneira estruturada e padronizada fornecida pelo *framework* de Extensões). Os plugins, por outro lado, fornecem novas funcionalidades ao oferecer suporte para tipos específicos de mídia nos navegadores. Um exemplo de extensão poderia ser uma que permite aos usuários salvar todas as abas abertas que não estão no modo anônimo no *local storage*. Um exemplo de plugin poderia ser um que permite a leitura e renderização de arquivos PDF no navegador.

De acordo com (TrueList, 2024), existem mais de 180.000 extensões disponíveis para o Google Chrome. As extensões mais famosas em 2024, de acordo com (DebugBear, 2024), são:

1. **Adblock Plus** - Ferramenta de bloqueio de anúncios;
2. **Grammarly for Chrome** - Assistente de escrita e gramática;
3. **Honey** - Extensão para encontrar e aplicar cupons de desconto automaticamente;
4. **Google Translate** - Tradução de páginas e textos;
5. **uBlock Origin** - Outra popular extensão de bloqueio de anúncios;
6. **LastPass** - Gerenciador de senhas;
7. **Evernote Web Clipper** - Ferramenta para salvar trechos da web diretamente no Evernote;
8. **Pinterest Save Button** - Permite salvar imagens e páginas diretamente no Pinterest;
9. **Avast Online Security** - Oferece segurança adicional durante a navegação;
10. **Cisco Webex** - Extensão para reuniões online.

2.4.1 Tecnologias para criar extensões

As extensões oferecem simplicidade aos desenvolvedores, sendo criadas com HTML, CSS, JavaScript e JSON. O HTML estrutura o conteúdo das páginas *web*, enquanto o CSS define a aparência, como cores e layout. JavaScript adiciona interatividade e funcionalidades dinâmicas, e o JSON é usado para armazenar e trocar dados de forma leve e organizada entre sistemas. Ademais, qualquer sistema operacional e um editor de texto podem ser utilizados para criar uma extensão, considerando que estas são apenas um conjunto de arquivos HTML e JavaScript.

Para implementar a lógica na extensão, utiliza-se o JavaScript, além de acessar APIs. Para criar a interface visual para o usuário final e complementar o JavaScript, são utilizados HTML e CSS. Por fim, o JSON é empregado para a criação do manifesto (manifesto.json), é um arquivo crucial que define as informações básicas da extensão, como nome, versão, permissões, ícones e recursos que a extensão utiliza.

Dessa forma, existem alguns passos para realizar o *upload* de uma extensão:

1. Deve-se criar uma pasta dedicada ao projeto em qualquer sistema operacional e editor de texto;
2. É necessário criar três arquivos. Os arquivos responsáveis pelo JavaScript, HTML e CSS podem ter nomes arbitrários, no entanto, o arquivo contendo o JSON deve obrigatoriamente se chamar *manifest.json*;
3. Para fazer o *upload* na página chrome://extensions/, é necessário compactar a pasta do projeto em um arquivo *zip* e carregá-lo na área do desenvolvedor. Essa área é uma seção específica do navegador que permite gerenciar, testar e depurar extensões, aplicativos e temas, facilitando o desenvolvimento e ajustes necessários.;
4. Por fim, basta ativar a extensão desenvolvida.

2.4.2 Segurança

É essencial entender que as extensões possuem a capacidade de executar funções que vão muito além das capacidades de um navegador comum. Portanto, é de suma importância ter uma atenção rigorosa à segurança, garantindo que o usuário não seja prejudicado ao utilizar a extensão.

Pode-se considerar o gerenciamento dos dados do usuário, onde salvar os dados no armazenamento pode ser configurado em uma API de *storage* fornecida pelas extensões, mas também pelo próprio *local storage*, que é uma forma de armazenamento *web* que permite que os navegadores salvem dados localmente no dispositivo do usuário. No entanto, isso apresenta desvantagens, pois os dados não são criptografados, sendo possível acessar com facilidade dados sensíveis.

Algumas informações pessoais para conexão com APIs externas não devem estar diretamente presentes no código, como, por exemplo, a chave de uma API. Além disso, é extremamente importante filtrar todos os dados sensíveis do usuário ao receber informações de uma API.

2.5 Flask

O conteúdo apresentado nesta seção baseia-se em (Copperwaite, 2015).

O Flask é um *framework* escrito em python que é bastante utilizado no mundo, pois é simples e rapido de ser criado. Essa ferramente disponibiliza inumeras bibliotecas poderosas que são responsáveis por suprir a necessidade de problemas comuns no desenvolvimento de softwares, como por exemplo (Copperwaite, 2015):

- Roteamento de URL, que facilita o mapeamento de URLs para o código, permitindo o gerenciamento organizado de diferentes partes do site e direcionando o usuário para o conteúdo correto com base na URL solicitada;
- Renderização de templates com Jinja2, que permite a combinação de HTML com lógica de programação para criar páginas dinâmicas e interativas de forma eficiente.
- Gerenciamento de sessões e segurança de cookies;
- Análise de requisições HTTP e manipulação flexível de respostas;
- Depurador interativo baseado na *web*;
- Gerenciamento de configuração de aplicação flexível e fácil de usar.

O uso desse *framework* será relevante quando a extensão do navegador Google Chrome acessar o modelo treinado. Isso é possível porque, nesse *framework* em Python, é possível utilizar a biblioteca joblib (Varoquaux; Grisel; Candes, 2024) para carregar o modelo treinado.

2.5.1 Ambiente

Para o projeto Flask, é recomendado criar um ambiente virtual para gerenciar os pacotes utilizados no projeto, como o *venv* (Python Packaging Authority, 2024a).

O *venv* permite criar ambientes virtuais isolados para gerenciar pacotes em diferentes projetos, evitando conflitos entre dependências. Com um ambiente virtual, é possível instalar pacotes específicos para cada projeto de forma segura, sem que eles interfiram em outros ambientes ou projetos. Com isso, o projeto torna-se mais estável e fácil de manter. (Python Packaging Authority, 2024b).

Outra consideração relevante é que a instalação de pacotes diretamente no sistema pode exigir privilégios de *root*. Com o uso de ambientes virtuais, é possível criar ambientes isolados de Python e instalar pacotes sem a necessidade de permissões administrativas. Isso é especialmente importante em máquinas compartilhadas, onde normalmente não é simples obter acesso de administrador.

Além disso, o *pip* (Python Packaging Authority, 2024c) desempenha um papel muito importante na gestão dos pacotes de referência instalados pelo Python, sendo utilizado para a instalação e atualização de pacotes.

2.5.2 Implementação do framework

Após a configuração do ambiente, é possível implementar um projeto em Flask de forma simples e prática com pouco código, conforme o exemplo fornecido por (Copperwaite, 2015):

```
1  from flask import Flask
2
3  app = Flask(__name__)
4
5  @app.route('/')
6  def index():
7      return 'Hello, Flask!'
8
9  @app.route('/hello/<name>')
10 def hello(name):
11     return 'Hello, %s' % name
12
13 if __name__ == '__main__':
14     app.run(debug=True)
```

Na primeira linha, é importado o pacote *flask* para a aplicação em Python. Em seguida, na terceira linha, é criada uma instância da classe *Flask*, onde `__name__` é uma variável especial que indica onde a aplicação está sendo executada.

Nas linhas 5 a 7, define-se uma URL raiz (’/’), associando-a à função *index()*. Esta função retorna uma string que será exibida no navegador quando a URL raiz for acessada. Em seguida, nas linhas 9 a 11, define-se uma rota GET que aceita o parâmetro *name* para exibir um texto que varia de acordo com o valor do parâmetro.

Por fim, na linha 13, verifica-se se o script está sendo executado diretamente e não como um módulo. Dessa forma, na linha 14, a aplicação pode ser executada em modo *debug*, permitindo a atualização automática e a exibição detalhada de erros.

3 Experimentos

A partir do modelo desenvolvido no artigo "Estudo e Aplicação de SVMs na Detecção de Fake-News" (Silva; Fagundes, 2021), foi selecionada a melhor abordagem indicada pelo trabalho para o retreinamento, para que dessa forma seja possível armazenar o modelo SVM. A abordagem escolhida foi a SVM com *kernel* polinomial, construída a partir de uma matriz *Bag of Words* (BoW) sem *stopwords*.

No início, o modelo SVM foi retreinado com o *Corpus FAKE.BR* (Monteiro *et al.*, 2018). No entanto, como esse conjunto foi atualizado até 2019, foi necessário escolher um banco de dados mais recente. Assim, o *FakeRecogna* (Garcia; Afonso; Papa, 2022) também foi selecionado. Dessa forma, o primeiro modelo foi treinado apenas com o *Corpus FAKE.BR*, o segundo somente com o *FakeRecogna*, e, por fim, o terceiro modelo foi treinado a partir da combinação dos dois bancos de dados.

Em seguida, o modelo gerado foi armazenado utilizando a biblioteca *joblib* (Varoquaux; Grisel; Candes, 2024), sendo posteriormente carregado em uma aplicação *back-end* desenvolvida em Python com o *framework* Flask (Copperwaite, 2015).

Posteriormente, para utilizar o modelo de forma mais simplificada, foi criada uma extensão para o Google Chrome, onde foi utilizada a linguagem *JavaScript* para capturar as informações e enviar os dados da notícia para a API.

Por fim, foram escolhidos três sites de notícias de empresas jornalísticas reconhecidas para realizar o teste: a Globo (Globo, 2024), o UOL (Online, 2024) e a CNN Brasil (Brasil, 2024). Quatro temáticas foram selecionadas:

- Economia;
- Política;
- Esporte;
- Saúde.

Cada item pode tratar de assuntos relacionados ao Brasil ou internacionais; no entanto, todas as notícias estão em língua portuguesa brasileira. O ano da notícia foi levado em consideração, e assim foi escolhida uma notícia de cada temática para cada ano, desde 2020 até 2024, totalizando 60 notícias testadas. Cada notícia foi testada com cada modelo.

As etapas da experimentação realizadas neste trabalho podem ser ilustradas conforme apresentado na Figura 3.1.

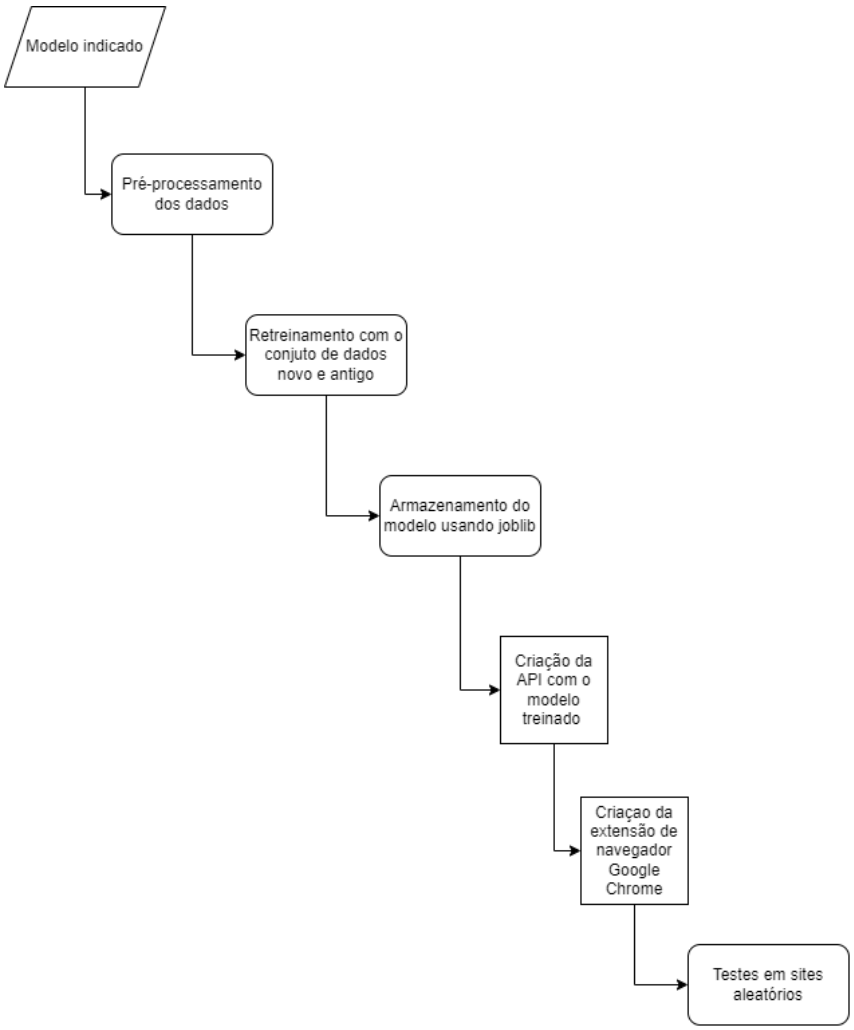

Figura 3.1 – Caminho realizado no experimento

3.1 Conjunto de dados

3.1.1 *Corpus FAKE.BR*

O *Corpus FAKE.BR* (Monteiro *et al.*, 2018), criado a partir de um trabalho realizado por Renato M. Silva, Roney L. S. Santos, Tiago A. Almeida e Thiago A. S. Pardo (Silva *et al.*, 2020), contém 7200 (sete mil e duzentas) notícias, todas em português, sendo 50% das notícias verdadeiras e 50% falsas[1].

De acordo com a Tabela 3.1, podemos observar alguns exemplos, onde temos o ID de cada notícia, a notícia referenciada no campo "Sample" e a "Label", sendo 1 para notícias

[1] O *dataset* pode ser acessado através do *link* https://github.com/roneysco/Fake.br-Corpus

falsas e 0 para as verdadeiras.

Tabela 3.1 – Dados do *Corpus FAKE.BR*

ID	Sample	Label
0	Kátia Abreu diz que vai colocar sua expulsão e...	1
1	Dr. Ray peita Bolsonaro, chama-o de “conservad...	1
2	Reinaldo Azevedo desmascarado pela Polícia Fed...	1
3	Relatório assustador do BNDES mostra dinheiro ...	1
4	Radialista americano fala sobre o PT: “Eles ve...	1
...	...	...
7195	Para jornal britânico, ação contra Lula na Lav...	0
7196	Temer diz que acionou PF e Cade para investiga...	0
7197	Os obstáculos políticos de Temer em 2017. Espe...	0
7198	Sexta-feira, 15 de setembro de 2017. Boa noit...	0
7199	‘Não me envolvo em política’, diz brasileiro q...	0

Para criar esse *dataset*, os autores pesquisaram na internet por notícias falsas e verificaram cada uma delas para garantir que realmente fossem falsas. Caso uma notícia fosse uma meia-verdade, ela era excluída do banco de dados. Por fim, automatizaram uma busca por versões verdadeiras correspondentes às notícias falsas. Um problema encontrado durante os testes foi que a última atualização desse *dataset* foi em 2019, o que pode resultar em falhas nos resultados para algumas notícias atuais.

Além disso, os autores comentam os pontos que foram utlizados para capturar as notícias falsas:

- Não há autor na notícia;
- As notícias contêm erros gramaticais;
- As notícias possuem títulos sensacionalistas;
- São utilizadas afirmações muito fortes nas notícias;
- As notícias fazem uso exagerado de pontos de exclamação e letras maiúsculas para chamar a atenção;

- A notícia não possui fonte.

3.1.2 *FakeRecogna*

O conjunto de dados *FakeRecogna*[2] foi o segundo *dataset* utilizado para o treinamento das SVMs, foi criado a partir de um trabalho realizado por Gabriel L. Garcia, Luis C. S. Afonso e João Paulo Papa (Garcia; Afonso; Papa, 2022). Os autores selecionaram e categorizaram 11.902 (onze mil novecentas e duas) notícias, todas em português, onde 5.951 (cinco mil novecentos e cinquenta e uma) são falsas e 5.951 (cinco mil novecentos e cinquenta e uma) são verdadeiras. Esse conjunto foi dividido da seguinte maneira:

- Título;
- Subtítulo;
- Corpo da notícia;
- Categoria da notícia;
- Autor da notícia;
- Data de publicação;
- *URL* da notícia;
- Classe da notícia.

Foi necessário considerar somente o corpo da notícia para que ficasse igual ao conjunto anterior. No entanto, foi necessário inverter o rótulo de classificação de cada notícia do *FakeRecogna* para se adequar ao padrão do conjunto anterior. Sendo assim, foi possível obter a Tabela 3.2.

[2] O *dataset* pode ser encontrado no *link* https://github.com/Gabriel-Lino-Garcia/FakeRecogna

Tabela 3.2 – Dados do *FakeRecogna*

ID	Sample	Label
0	Papa Francisco foi preso sob acusação de tráfico humano e ...	1
1	Equador prepara cova coletiva para mortos por COVID-19	0
2	Air France voltará a operar voo direto Pequim-Paris	0
3	Marfrig intensifica venda de carne do Brasil para a China	0
4	As parciais das eleições de 2014 alternaram maioria durante apuração	1
...	...	...
11898	Argentina vacinou duas vezes o mesmo homem contra COVID-19	1
11899	Ministro se encontra com representantes da ONU	0
11900	video mostrar garoto afogar ninguem nado	1
11901	Toque de recolher na França a partir das 20h	0
11902	Sistema usado em video para simular fraude nao funciona	1

Os passos para a criação deste dataset foram os seguintes:

1. A coleta de notícias foi realizada por bots de rastreamento (*crawlers*) desenvolvidos para minerar páginas de agências de notícias conhecidas e de grande importância nacional. A mineração de notícias falsas foi focada principalmente nas páginas mencionadas pelo (Lab, 2024);
2. Segundo, foi realizada uma coleta de notícias verdadeiras em sites reconhecidos e confiáveis, como UOL, G1 e Extra. Foram coletadas mais de 100.000 (cem mil) amostras, das quais 5.951 (cinco mil novecentos e cinquenta e uma) foram filtradas para manter o equilíbrio;
3. Terceiro, o dataset foi estruturado em oito colunas, contendo diferentes informações sobre cada notícia;
4. Quarto, cada notícia foi organizada com uma classificação: 0 para notícias falsas e 1 para notícias verdadeiras. Entretanto, essa classificação foi invertida para se adequar ao conjunto anterior, como já mencionado.

Diferente do *Corpus FAKE.BR*, este *dataset* teve notícias falsas e notícias verdadeiras totalmente desconexas. Enquanto o conjunto anterior procurava uma notícia falsa e, em seguida, buscava uma notícia verdadeira correspondente, este conjunto selecionou cada notícia de forma aleatória. Foi descoberto que essa metodologia afetou um pouco a acurácia do modelo treinado com o conjunto do *FakeRecogna*.

É possível visualizar os métodos usados para preenchimento do banco de dados na Figura 3.2.

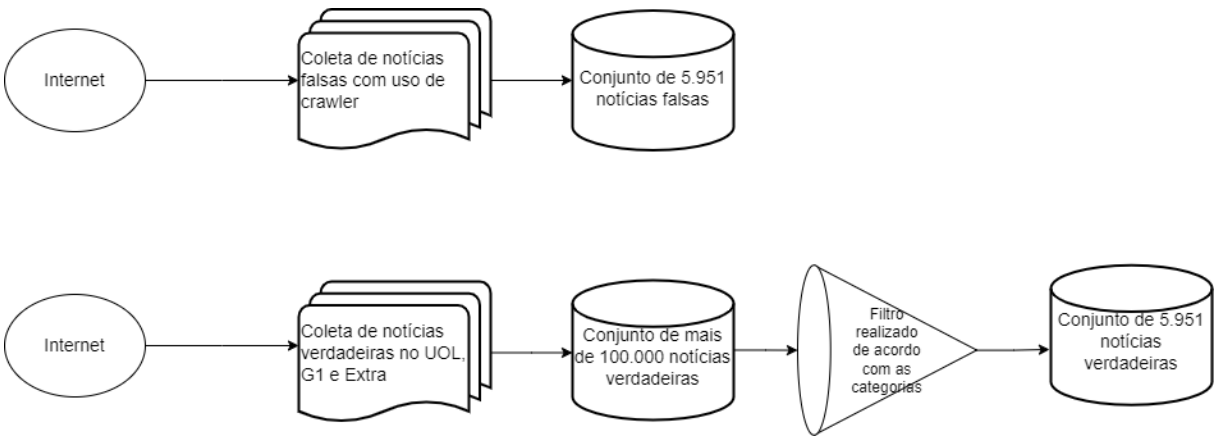

Figura 3.2 – Método utilizado pelo *FakeRecogna* para popular o banco de dados

O resultado final ficou armazenado em uma pasta XLSX contendo as notícias em oito colunas diferentes. Cada uma foi lida e o valor da classe foi invertido. O resultado final foi armazenado em um arquivo PKL no mesmo formato que o modelo usa com o *Corpus FAKE.BR*.

3.1.3 Criação de um banco de dados unificado

A unificação dos dois conjuntos, *FakeRecogna* e *Corpus FAKE.BR*, resultou em um novo conjunto. O conjunto *FakeRecogna* , composto por 11.902 elementos, foi combinado com o *Corpus FAKE.BR*, que continha 7.200 elementos. Dessa forma, o terceiro conjunto totaliza 19.102 elementos.

3.2 Pré-processamento e Treinamento

De acordo com (Silva; Fagundes, 2021) a melhor abordagem para treinar o modelo seria o *kernel* polinomial, que seria encontrar a similaridade por meio do cálculo de um polinômio, além disso foi usada a BoW para transformação dos dados e também foram removidos os *stopwords*. Isso tudo foi baseado em uma métrica chamada *F1 score*, que é uma métrica de avaliação que combina a precisão e a sensibilidade de um modelo em uma única métrica. Pode-se analisar o resultado final na Tabela 3.3, retirada de (Silva; Fagundes, 2021).

Tabela 3.3 – Tabela de resultados dos melhores modelos para BoW, com remoção de stopwords

BoW + stopwords	**F1**	**Precisao**	**Sensibilidade**	**Acuracia**
Linear	0.9632	0.9640	0.9626	0.9631
RBF	0.9624	0.9636	0.9612	0.9623
Poly	0.9648	0.9599	0.9698	0.9645

3.2.1 *Bag of Words*

Bag of Words (BoW) é uma técnica de extração de características amplamente utilizada para modelar dados textuais, aplicada em algoritmos de recuperação de informação e aprendizado de máquina. Especificamente, os modelos BoW representam um conjunto não estruturado de todas as palavras conhecidas em um documento de texto, definidas exclusivamente pela frequência, sem considerar a ordem ou o contexto das palavras (Murel; Kavlakoglu, 2024). Essa abordagem é uma etapa comum em diversos processos de mineração de texto, contribuindo significativamente para a preparação dos dados para análise posterior.

A compreensão da extração de características por BoW exige, ao menos, um entendimento básico de espaços vetoriais. Um espaço vetorial é um espaço multidimensional no qual pontos são plotados. No contexto da abordagem Bag of Words, cada palavra individual torna-se uma dimensão (ou eixo) do espaço vetorial. Se um conjunto de textos possui n palavras distintas, o espaço vetorial resultante terá n dimensões, uma para cada palavra única presente no conjunto de textos. O modelo, então, representa cada documento de texto como um ponto dentro deste espaço vetorial. A posição de um ponto ao longo de uma determinada dimensão é determinada pela frequência de ocorrência da palavra correspondente àquela dimensão no documento representado pelo ponto (Murel; Kavlakoglu, 2024).

Podem ser utilizados como exemplos os seguintes textos extraídos de (Murel; Kavlakoglu, 2024):

1. *A rose is red, a violet is blue.*
2. *My love is like a red, red rose.*

Um espaço vetorial para um corpus contendo esses dois documentos teria dimensões separadas para as palavras *"red"*, *"rose"* e *"violet"*. Um espaço vetorial tridimensional para essas palavras é representado pela Figura 3.3.

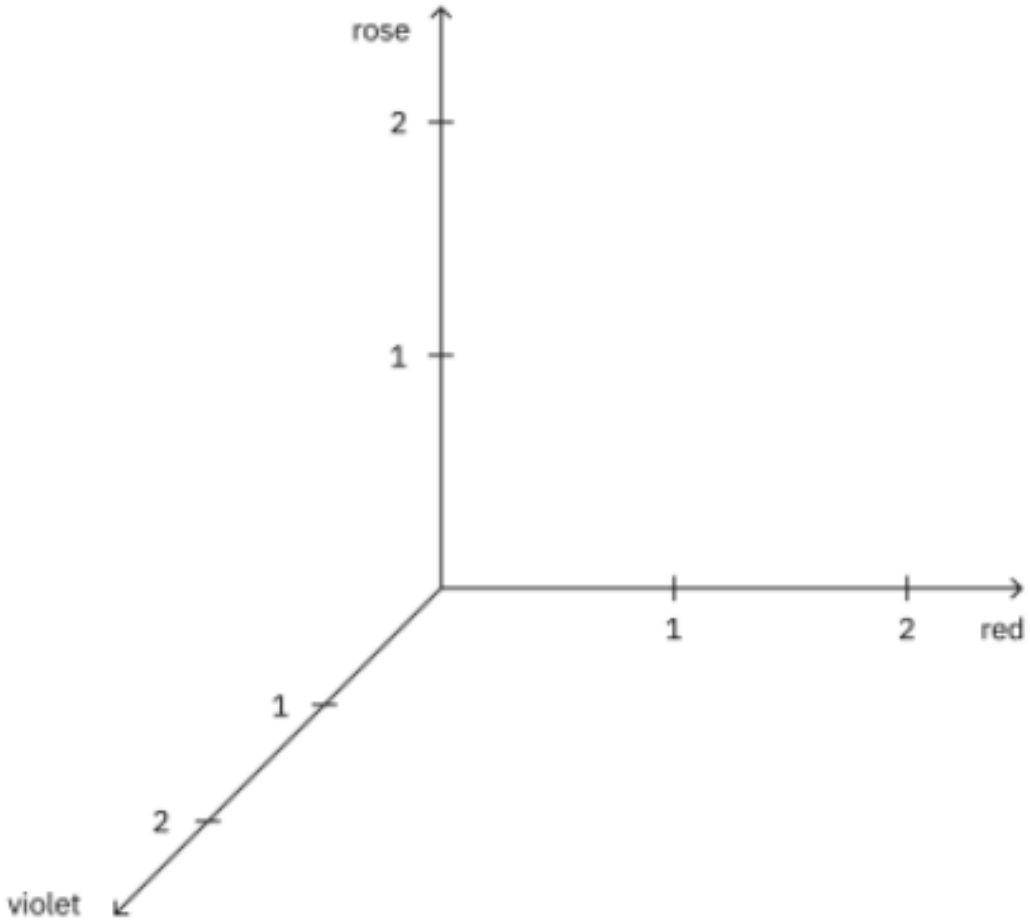

Figura 3.3 – Gráfico tridimensional com palavras "*rose*", "*red*", e "*violet*"como dimensões retirado de (Murel; Kavlakoglu, 2024)

Como as palavras *"rose"*, *"red"*, e *"violet"*ocorrem uma vez no primeiro exemplo, o vetor correspondente a esse documento neste espaço vetorial é representado por (1,1,1). No segundo exemplo, a palavra *"red"*aparece duas vezes, *"rose"*uma vez, e *"violet"*não aparece. Assim, o ponto vetorial para o segundo exemplo é (2,1,0). Esses pontos correspondentes aos documentos são mapeados no espaço vetorial tridimensional de acordo com a Figura 3.4

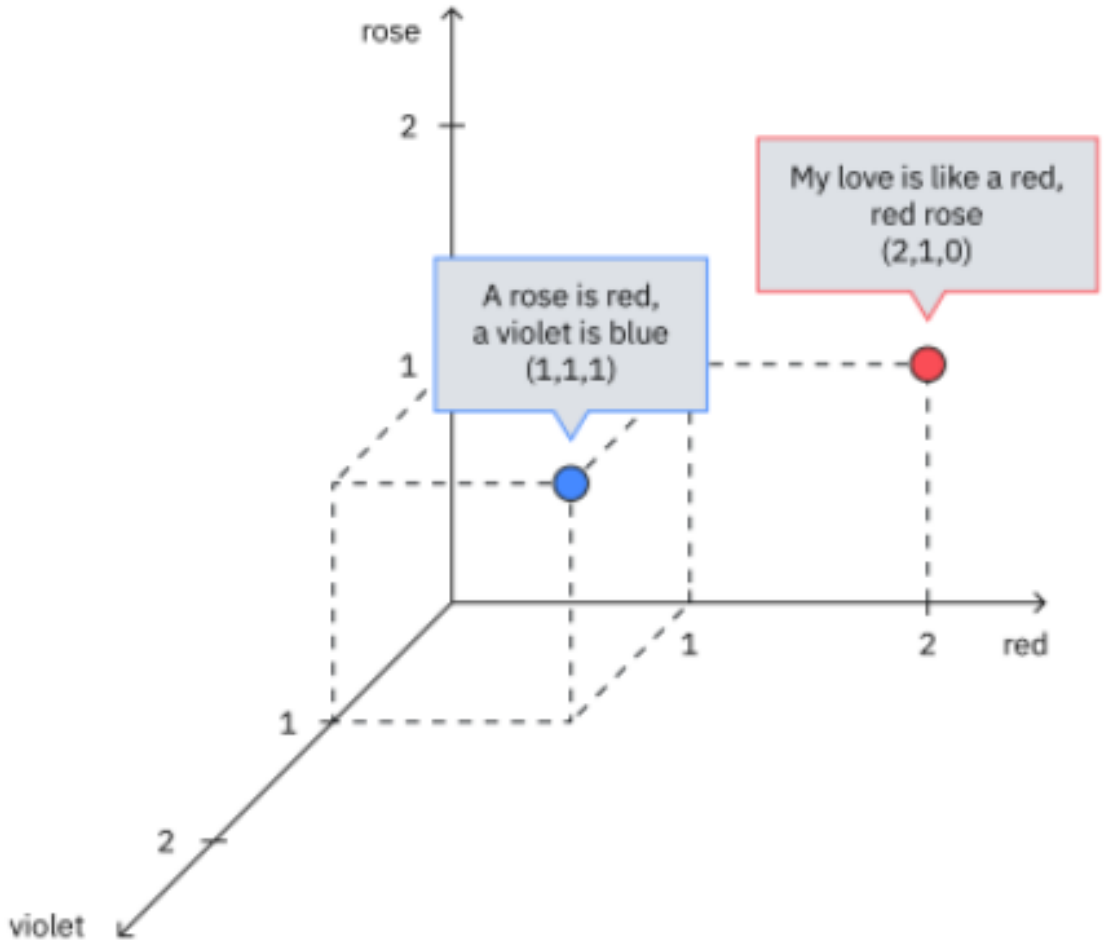

Figura 3.4 – Representação vetorial dos documentos no espaço tridimensional com base nas frequências das palavras *"rose"*, *"red"*, e *"violet"*retirado de (Murel; Kavlakoglu, 2024)

Portanto, para o BoW o que realmente é apenas a frequência com que as palavras aparecem nos exemplos, desconsiderando a ordem ou o contexto em que ocorrem. Os procedimentos iniciais para a utilização do BoW envolvem a limpeza dos dados e a tokenização. A limpeza é realizada para padronizar o texto, incluindo a remoção de acentos, símbolos e outros elementos indesejados. A tokenização tem como objetivo dividir o texto em unidades menores, como palavras ou termos, permitindo que sejam processados de forma eficiente. Após a limpeza e tokenização, os dados são processados utilizando o método BoW, incorporando o uso de *stopwords*.

3.2.2 Limpeza de dados

Antes de treinar e usar o modelo, é necessário realizar um pré-processamento do texto. As letras são convertidas para a forma minúscula e os acentos são retirados. Também são removidos a pontuação, caracteres especiais e espaços em branco extras. Além disso, os dígitos são substituídos por '0', as *URLs* por 'URL' e as páginas de redes sociais por 'REDESOCIAL'.

Isso é importante para o processamento pelo computador, que entende cada caractere como um código binário. Dessa forma, cada palavra é diferente. Mesmo que tenha o mesmo sentido, se um caractere for maiúsculo ou tiver um acento a mais, para o computador já é um valor diferente. Dessa forma, foi muito importante para o treinamento do modelo e para usar o modelo posteriormente nas notícias encontradas nos sites escolhidos. Um exemplo de notícia após a limpeza pode ser visto na Tabela 3.4

Tabela 3.4 – Exmplo de pré-processamento

Notícia no site	**Notícia após o pré-processamento**
Os investimentos diretos no país (IDP) registraram ingressos líquidos de US$ 9,6 bilhões em março de 2024, ante US$ 7,3 bilhões em março de 2023, informou o Banco Central nesta quinta-feira (2). É o melhor resultado para o mês desde 2012, quando o país recebeu ingressos líquidos de US$ 14,9 bilhões, segundo a série histórica do BC, iniciada em 1995. O número também é o melhor desde agosto de 2022. Na época, o IDP era de R$ 9,7 bilhões. No documento estatísticas do setor externo, a autoridade monetária aponta que o número de março foi derivado de ingressos líquidos de US$ 4,1 bilhões em participação no capital e de US$ 5,5 bilhões em operações intercompanhia. O IDP acumulado em doze meses totalizou US$ 66,5 bilhões (2,98% do PIB) em março de 2024, ante US$ 64,3 bilhões (2,90% do PIB) no mês anterior e US$ 75,3 bilhões (3,76% do PIB) em relação a março de 2023. De acordo com o BC, os investimentos na bolsa de valores brasileira foram equilibrados em março de 2024, com as despesas superando ligeiramente as receitas, resultando em saídas líquidas de US$72 milhões. A autoridade monetária registrou ainda saídas líquidas de US$ 3,3 bilhões em ações e fundos de investimento, que foram compensados por ingressos na ordem de US$3,2 bilhões em títulos de dívida. Nos últimos doze meses, os investimentos em carteira no mercado doméstico somaram ingressos líquidos de US $6,3 bilhões.	os investimentos diretos no pais idp registraram ingressos liquidos de us 0 bilhoes em marco de 0 ante us 0 bilhoes em marco de 0 informou o banco central nesta quintafeira 0 e o melhor resultado para o mes desde 0 quando o pais recebeu ingressos liquidos de us 0 bilhoes segundo a serie historica do bc iniciada em 0 o numero tambem e o melhor desde agosto de 0 na epoca o idp era de r 0 bilhoes no documento estatisticas do setor externo a autoridade monetaria aponta que o numero de marco foi derivado de ingressos liquidos de us 0 bilhoes em participacao no capital e de us 0 bilhoes em operacoes intercompanhia o idp acumulado em doze meses totalizou us 0 bilhoes 0 do pib em marco de 0 ante us 0 bilhoes 0 do pib no mes anterior e us 0 bilhoes 0 do pib em relacao a marco de 0 leia mais de acordo com o bc os investimentos na bolsa de valores brasileira foram equilibrados em marco de 0 com as despesas superando ligeiramente as receitas resultando em saidas liquidas de us 0 milhoes a autoridade monetaria registrou ainda saidas liquidas de us 0 bilhoes em acoes e fundos de investimento que foram compensados por ingressos na ordem de us 0 bilhoes em titulos de divida nos ultimos doze meses os investimentos em carteira no mercado domestico somaram ingressos liquidos de us 0 bilhoes

3.2.3 Tokenização

A tokenização é o processo responsável por dividir o texto em unidades menores, chamadas de *tokens*, que podem ser interpretadas pelo modelo. Optou-se por usar palavras em vez de frases ou blocos maiores de texto devido ao maior grau de especificidade que as palavras individuais oferecem. O objetivo é identificar palavras que aumentem a probabilidade de uma notícia ser classificada como falsa ou verdadeira.

Para identificar os *tokens*, o algoritmo utiliza espaços em branco e pontuações como critérios de separação. No entanto, como os dados já foram previamente processados e limpos, apenas os espaços em branco serão utilizados para determinar onde as separações devem

ocorrer. É possível visualizar um exemplo na Tabela 3.5

Tabela 3.5 – Exemplo de tokenização

Notícia antes da tokenização	**Notícia após a tokenização**
tragedia anunciada reporter alertou sobre queda de camera na vila olimpica ninguem tomou atitude	[’tragedia’, ’anunciada’, ’reporter’, ’alertou’, ’sobre’, ’queda’, ’de’, ’camera’, ’na’, ’vila’, ’olimpica’, ’ninguem’, ’tomou’, ’atitude’]

3.2.4 Remoção de *stopwords*

O objetivo de remover as *stopwords* do vocabulário é simplificar o conjunto de dados sem causar perdas significativas no desempenho do classificador. Para isso, foram selecionadas palavras muito frequentes da língua portuguesa em um contexto geral, pois suas funções gramaticais as tornam indispensáveis em qualquer tipo de texto, independentemente de sua natureza. Portanto, é improvável que, se consideradas como atributos, essas palavras tenham relevância para diferenciar as classes. As *stopwords* escolhidas para serem removidas do vocabulário foram fornecidas pela biblioteca NLTK(Project, 2024). É possível listar as principais classes gramaticais que atendem aos requisitos, a saber:

- Artigos;
- Preposições;
- Pronomes;
- Advérbios;
- Verbos auxiliares.

Dessa forma, é possível visualizar todas as *stopwords* que foram removidas durante o treinamento e a utilização do modelo, conforme a Tabela 3.6

Tabela 3.6 – Lista de *stopwords*

['de', 'a', 'o', 'que', 'e', 'é', 'do', 'da', 'em', 'um', 'para', 'com', 'não', 'uma', 'os', 'no', 'se', 'na', 'por', 'mais', 'as', 'dos', 'como', 'mas', 'ao', 'ele', 'das', 'à', 'seu', 'sua', 'ou', 'quando', 'muito', 'nos', 'já', 'eu', 'também', 'só', 'pelo', 'pela', 'até', 'isso', 'ela', 'entre', 'depois', 'sem', 'mesmo', 'aos', 'seus', 'quem', 'nas', 'me', 'esse', 'eles', 'você', 'essa', 'num', 'nem', 'suas', 'meu', 'às', 'minha', 'numa', 'pelos', 'elas', 'qual', 'nós', 'lhe', 'deles', 'essas', 'esses', 'pelas', 'este', 'dele', 'tu', 'te', 'vocês', 'vos', 'lhes', 'meus', 'minhas', 'teu', 'tua', 'teus', 'tuas', 'nosso', 'nossa', 'nossos', 'nossas', 'dela', 'delas', 'esta', 'estes', 'estas', 'aquele', 'aquela', 'aqueles', 'aquelas', 'isto', 'aquilo', 'estou', 'está', 'estamos', 'estão', 'estive', 'esteve', 'estivemos', 'estiveram', 'estava', 'estávamos', 'estavam', 'estivera', 'estivéramos', 'esteja', 'estejamos', 'estejam', 'estivesse', 'estivéssemos', 'estivessem', 'estiver', 'estivermos', 'estiverem', 'hei', 'há', 'havemos', 'hão', 'houve', 'houvemos', 'houveram', 'houvera', 'houvéramos', 'haja', 'hajamos', 'hajam', 'houvesse', 'houvéssemos', 'houvessem', 'houver', 'houvermos', 'houverem', 'houverei', 'houverá', 'houveremos', 'houverão', 'houveria', 'houveríamos', 'houveriam', 'sou', 'somos', 'são', 'era', 'éramos', 'eram', 'fui', 'foi', 'fomos', 'foram', 'fora', 'fôramos', 'seja', 'sejamos', 'sejam', 'fosse', 'fôssemos', 'fossem', 'for', 'formos', 'forem', 'serei', 'será', 'seremos', 'serão', 'seria', 'seríamos', 'seriam', 'tenho', 'tem', 'temos', 'tém', 'tinha', 'tínhamos', 'tinham', 'tive', 'teve', 'tivemos', 'tiveram', 'tivera', 'tivéramos', 'tenha', 'tenhamos', 'tenham', 'tivesse', 'tivéssemos', 'tivessem', 'tiver', 'tivermos', 'tiverem', 'terei', 'terá', 'teremos', 'terão', 'teria', 'teríamos', 'teriam']

3.2.5 Treinamento e uso do modelo

Após o tratamento dos dados iniciais e a geração das matrizes de dados de entrada, a próxima etapa é utilizá-las para treinar um modelo de aprendizado de máquina, tornando-o capaz de classificar notícias como falsas ou verdadeiras. A abordagem de aprendizado de máquina escolhida foi a de Máquinas de Vetores de Suporte (SVM), que realiza a classificação procurando por hiperplanos em um espaço multidimensional que distinguem claramente as classes.

Conforme descrito no estudo de (Silva; Fagundes, 2021), o modelo indicado com o melhor *F1-score* foi treinado com o *kernel* polinomial, utilizando o método BoW com remoção de *stopwords*. Os parâmetros configurados foram: C ajustado para 0,2, regulando o equilíbrio entre maximizar a margem de separação e minimizar erros de classificação; *gamma* em 0,01, controlando a influência das amostras de treinamento na SVM; *coef0* definido como 10, regulando a influência do termo independente; e *degree* igual a 3, especificando o grau do polinômio, determinando a complexidade da fronteira de decisão. Essa configuração ficou conforme a Tabela 3.7

Tabela 3.7 – Parâmetros do modelo BoW + stopwords

BoW + stopwords	**C**	*gamma*	*coef0*	*degree*
Poly	0.2	0.01	10.0	3

Após o retreinamento, o modelo foi salvo utilizando a biblioteca Joblib (Varoquaux; Grisel; Candes, 2024). Essa biblioteca permite a serialização de *arrays do NumPy* e, consequentemente, do modelo treinado, o que facilita seu armazenamento e posterior acesso para reutilizações ou validações futuras. O processo de serialização é responsável por converter o objeto em uma sequência de bytes que pode ser armazenada ou transmitida e, posteriormente, desserializada para recriar o objeto original. O resultado final é um arquivo compactado com extensão *.joblib*. Posteriormente, esse arquivo pode ser carregado para uso em outra aplicação.

3.3 Aplicação

3.3.1 *Back-end*

Para consultar o modelo de forma eficiente, foi necessário desenvolver uma aplicação em Python utilizando o framework Flask (Copperwaite, 2015). A aplicação recebe uma requisição POST via API, cuja entrada é o texto da notícia, e retorna uma mensagem com a estimativa probabilidade de ser uma notícia falsa.

Durante essa requisição, o texto é pré-processado e, em seguida, o modelo é carregado utilizando a biblioteca *joblib*. Dessa forma, o texto passa pelo modelo, que retorna uma resposta em forma de porcentagem. Esse processo pode ser visualizado na figura 3.5.

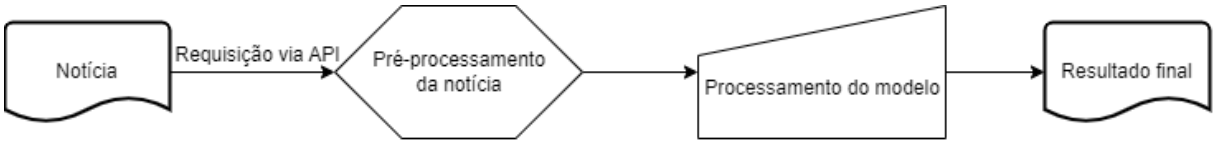

Figura 3.5 – Estrutura da API

De acordo com o código do arquivo app.py a seguir[3], é possível analisar os passos na Figura 3.5. O código foi simplificado para o uso do modelo treinado com o *FakeRecogna*.

```
1  from flask import Flask, request, render_template
2  from joblib import load
3  from preprocessing import preprocessing
4
5  app = Flask(__name__)
6
7  model = load('modelo_treinado_fake_recogna.joblib')
8
9  @app.route('/', methods=['GET', 'POST'])
10 def home():
11     if request.method == 'POST':
12
13         text = request.form['text']
14
15         pre_processed_text = preprocessing(text)
16
17         prediction = model.predict_proba([pre_processed_text])
18
19         first_row_percentages = [f"{prob * 100:.0f}%" for prob
               in prediction[0]]
20
21         return {"Chance de ser fake news":
               first_row_percentages[1]}
22
23 if __name__ == '__main__':
24     app.run(debug=True)
```

Na linha 7, o modelo treinado é importado para o código. Na linha 9, a rota de acesso à aplicação é criada para requisições *GET* e *POST*. Na linha 10, a função *home* é definida, onde, se o método for do tipo *POST*, o parâmetro *text* é utilizado para definir a variável *text*. Em seguida, o código pega o texto enviado e o pré-processa na linha 15. Já na linha 17, o texto pré-processado é passado para o modelo treinado, que retorna uma resposta em porcentagem. Na linha 19, a resposta é convertida em porcentagem sem casa decimal. Por fim, na linha 21, a resposta é retornada. As linhas 23 e 24 são responsáveis por manter a aplicação *Flask* rodando na porta 5000.

[3] O código pode ser acessado através do *link* https://github.com/lucas-19/fake-news-plugin-backend.

3.3.2 *Front-end*

O objetivo da criação da extensão para o Google Chrome é disponibilizar o modelo ao usuário final. A extensão foi desenvolvida em JavaScript, permitindo que o usuário capture informações ao selecionar o texto e clicar com o botão auxiliar do mouse. Após o desenvolvimento da aplicação, foi necessário realizar o upload no endereço ‘chrome://extensions/‘. Isso exibirá a versão da aplicação, o nome da aplicação e uma breve descrição, todas definidas no arquivo *manifest.json*, como pode ser visto na Figura 3.6

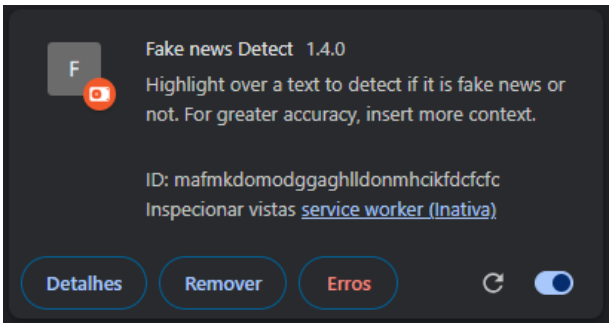

Figura 3.6 – Extensão para detectar *fake-news*

O texto selecionado é então enviado para a API, que retorna a resposta final exibida na tela do usuário. As etapas do processo consistem em:

1. Selecionar o texto da notícia.
2. Clicar com o botão auxiliar do mouse para utilizar a extensão.
3. A resposta final é exibida na tela em forma de alerta.

É possível visualizar essas etapas na Figura 3.7, Figura 3.8 e Figura 3.9.

Figura 3.7 – Selecionar texto da notícia

O arquivo JSON[4]apresentado a seguir corresponde ao manifesto da extensão. No campo `"name"`, a aplicação é intitulada como *Fake news Detect*. A versão do manifesto está

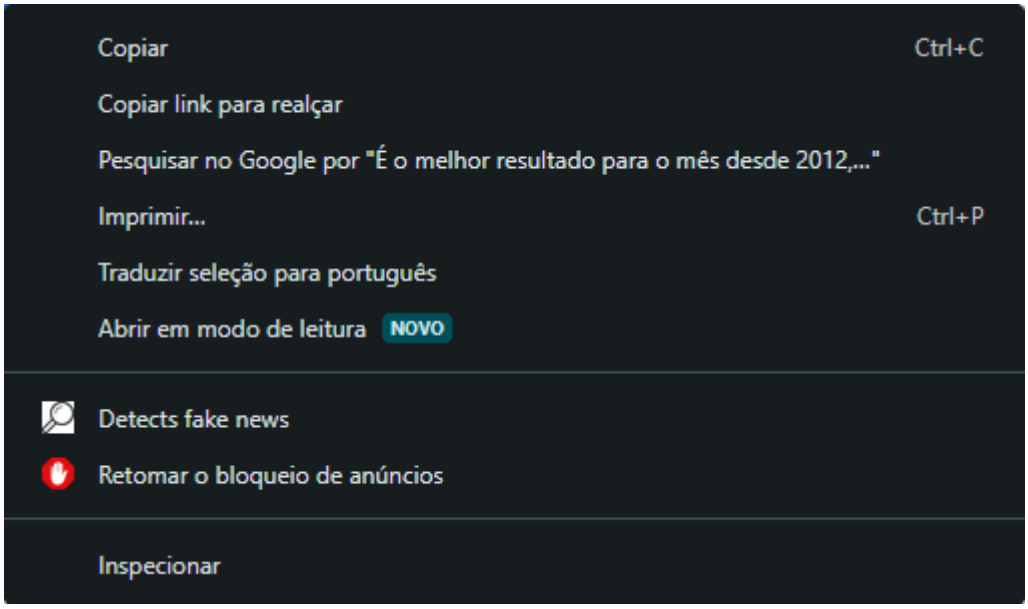

Figura 3.8 – Clicar com o botão auxiliar

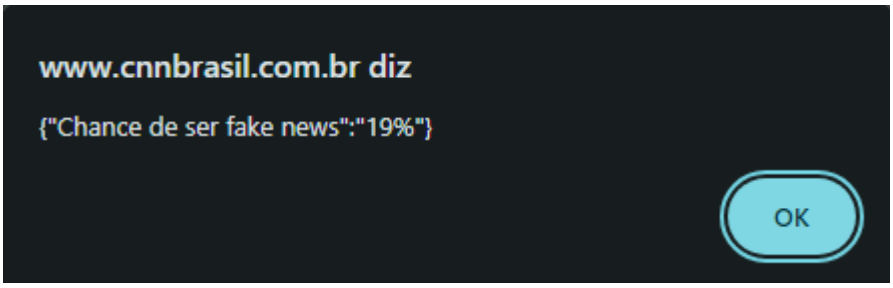

Figura 3.9 – Resultado final em formato de alerta

configurada como 3 e utiliza o arquivo `"background.js"` para a execução de tarefas em segundo plano. Além disso, o manifesto especifica permissões essenciais como o acesso ao menu de contexto (`"contextMenus"`), execução de scripts (`"scripting"`) e manipulação de abas ativas (`"activeTab"`). A configuração dos ícones também está presente, com a definição de um ícone de 16x16 pixels. O campo `"host_permissions"` autoriza a comunicação com o servidor local, através do endereço `http://127.0.0.1:5000/`.

[4] O arquivo JSON foi retirado de https://github.com/lucas-19/fake-news-plugin-front

```
{
    "name": "Fake news Detect",
    "description": "Highlight over a text to detect if it is
       fake news or not. For greater accuracy, insert more
       context.",
    "version": "1.3.3",
    "manifest_version": 3,
    "background": {
        "service_worker": "background.js"
    },
    "permissions": [
        "contextMenus",
        "scripting",
        "activeTab"
    ],
    "icons": {
        "16": "16.png"
    },
    "host_permissions": [
        "http://127.0.0.1:5000/"
    ]
}
```

No código a seguir, é apresentado o `background.js` que cria um menu de contexto chamado "Detects fake news", disponível quando o usuário seleciona um texto na página. A linha `chrome.runtime.onInstalled.addListener()` define um ouvinte que executa a função logo após a instalação da extensão. A função `chrome.contextMenus.create()` cria um item no menu de contexto, identificado pelo ID `"detectFakeNews"` e com o título *"Detects fake news"*, visível quando o usuário seleciona texto na página.

Quando o usuário clica no item do menu de contexto, um ouvinte é acionado, verificando se o item selecionado possui o ID `"detectFakeNews"`. Se confirmado, o texto selecionado pelo usuário é capturado e armazenado na variável `query`. Esse texto é então enviado para um servidor local, em `"http://127.0.0.1:5000"`, por meio de uma requisição HTTP POST, utilizando o método `fetch()`. O corpo da requisição é um objeto `FormData`, que encapsula o texto selecionado.

Quando a resposta da API *Flask* é recebida, ela é convertida para o formato JSON por meio de `response.json()`, e a função anônima subsequente processa essa resposta. A API `chrome.scripting.executeScript()` é então utilizada para injetar e executar um

script na aba atual do navegador. O script exibe a resposta do servidor como um alerta, usando `alert(JSON.stringify(data))`. Além disso, uma função auxiliar, `showAlert()`, é definida para transformar a resposta recebida em uma string JSON e exibi-la em um alerta. Por fim, um bloco `catch()` captura eventuais erros na requisição e os exibe no console do desenvolvedor, através de `console.error()`.

```
chrome.runtime.onInstalled.addListener(() => {
    chrome.contextMenus.create({
        id: "detectFakeNews",
        title: "Detects fake news",
        contexts: ["selection"]
    });
    alert('Context Menu Created!');
});

chrome.contextMenus.onClicked.addListener((info, tab) => {
    if (info.menuItemId === "detectFakeNews") {
        const query = info.selectionText;
        const url = "http://127.0.0.1:5000";
        const formData = new FormData();
        formData.append('text', query);

        fetch(url, {
            method: 'POST',
            body: formData
        })
        .then(response => response.json())
        .then(data => {
            chrome.scripting.executeScript({
                target: {tabId: tab.id},
                func: function(data) {
                    alert(JSON.stringify(data));
                },
                args: [data]
            });
        })
        .catch(error => {
            console.error('Error:', error);
        });
    }
```

```
});

function showAlert(data) {
    alert(JSON.stringify(data));
}
```

4 Resultados

Para analisar os resultados e comparar o desempenho entre os principais bancos de dados, além de avaliar a viabilidade da implementação desse modelo em situações reais, foi realizado um estudo com em três sites distintos. Foram analisadas 20 notícias de cada site, abrangendo quatro categorias temáticas: economia, política, esportes e saúde.

4.1 Uso da Ferramenta em Notícias Selecionadas

Para avaliar a utilização e a eficácia da aplicação, foi necessário capturar o corpo de texto de 60 notícias, distribuídas entre três veículos de comunicação:

- G1;
- UOL;
- CNN.

Dentro de cada veículo de comunicação, foi necessário coletar notícias dos anos de 2020 a 2024. Adicionalmente, de cada veículo foram selecionadas notícias pertencentes às seguintes categorias:

- Política;
- Esportes;
- Economia;
- Saúde.

Essas categorias foram selecionadas devido à diversidade de temas presentes em cada conjunto de dados, os quais se alinham com esses tópicos. Por exemplo, no *FakeBR Corpus*, as notícias estão distribuídas entre essas categorias, como evidenciado na Tabela 4.1.

Tabela 4.1 – Frequência de Artigos por Categoria (Silva *et al.*, 2020)

Categoria	**Frequência**
Economia	44
Ciência & Tecnologia	112
Sociedade & Notícias Diárias	1.276
Política	4.180
Religião	44
TV & Celebridades	1.544

.

Além disso, alguns desses tópicos também estão presentes no *FakeRecogna*, como mostra a Tabela 4.2

Tabela 4.2 – Categorias distribuidas do *Dataset FakeRecogna* (Garcia; Afonso; Papa, 2022)

Categoria	**# Notícias**	**%**
Brasil	904	7,6
Entretenimento	1.409	12,00
Saúde	4.456	37,4
Política	3.951	33,1
Ciência	602	5,1
Mundo	580	4,9
Total	**11.902**	**100,00**

Por fim, o conteúdo de cada notícia foi extraído e submetido à consulta na API, utilizando-se os modelos previamente treinados com os respectivos conjuntos de dados. O primeiro modelo foi treinado com o conjunto de dados do *Corpus FAKE.BR*, o segundo com o *FakeRecogna*, e o terceiro com a combinação de ambos. Dessa forma, foram realizados 180 testes no total, cujos resultados estão distribuídos na Tabela 4.3, Tabela 4.4 e Tabela 4.5 que apresentam os links para as notícias selecionadas.

Tabela 4.3 – Notícias retiradas da CNN

Ano	**Economia**	**Política**	**Esportes**	**Saúde**
2024	investimento estrangeiro no...	fab revoga documento...	santos chega a...	como cuidar do...
2023	fazenda ja ve...	brasil vai reavaliar...	juventus perde 10...	estudo descobre medicamento...
2022	poupanca tem valorizacao...	apoios no segundo...	ituano e vasco...	apoio de familiares...
2021	termeletricas a gas...	aecio neves nao...	comissao de etica...	vacina da astrazeneca...
2020	governo anuncia reformulacao...	coronel armando defende...	brasil na olimpiada...	medidas de biden...

Tabela 4.4 – Notícias retiradas da Globo

Ano	**Economia**	**Política**	**Esportes**	**Saúde**
2024	decisao fed efeitos...	comunicador celio alves...	projeto social da...	saude prorroga vacinacao...
2023	economia circular projeto...	nunes marques pede...	torneio agro de...	ministerio da saude...
2022	instituicoes debatem arranjos...	joao azevedo se...	quando a natureza...	levando saude atraves...
2021	seguro residencial sinonimo...	por que se...	o esporte de...	a importancia da...
2020	economia qual o...	gra bretanha endurecera...	governo publica criterios...	saude mental e...

Tabela 4.5 – Notícias retiradas do UOL

Ano	Economia	Política	Esporte	Saúde
2024	ocde eleva para...	quanto cada partido...	brasil cai no...	pela primeira vez...
2023	brasil lista maiores...	retrospectiva 2023 fatos...	confira as principais...	einstein um terco...
2022	crescimento de 21...	lula promete retorno...	surpresas do esporte...	dia mundial da...
2021	pib primeiro trimestre...	por que bolsonaro...	modalidades das olimpiadas...	oms brasil foi...
2020	governo corta a...	brigas travam reformas...	alemao ufc o...	7 em cada...

4.2 Análise dos resultados

Após a realização de todos os testes, foi possível armazenar e organizar os resultados obtidos para cada notícia, dividindo-os nas categorias já mencionadas. Nos gráficos construídos, o "Modelo 1" refere-se ao conjunto de dados do *FakeBR Corpus*, o "Modelo 2" ao *FakeRecogna*, e o "Modelo 3" à combinação de ambos os conjuntos de dados. Além disso, considerou-se que o conteúdo de cada notícia é totalmente verdadeiro.

4.2.1 Política

Na categoria política, foi possível observar que, em cada site analisado, os Modelos 1 e 2 obtiveram resultados satisfatórios. No entanto, é importante destacar que o conjunto de dados do *FakeBR Corpus* (Modelo 1) contém 4.180 notícias na categoria "Política", enquanto o conjunto de dados do *FakeRecogna* (Modelo 2) possui 3.951 notícias na mesma categoria. Verifica-se que, na categoria política, o Modelo 1 apresentou melhores resultados, em grande parte devido à estratégia de formação do conjunto de dados, onde cada notícia falsa é pareada com uma notícia verdadeira, bem como à maior quantidade de notícias disponíveis para treinamento.

Além disso, observou-se que as notícias de 2024 dos sites UOL e CNN apresentam uma menor quantidade de caracteres em comparação com a notícia da Globo. Especificamente, a notícia da UOL possui 1.323 caracteres, a da CNN têm 2.778 caracteres, enquanto a da Globo possui 3.598 caracteres. Essa variação no tamanho dos textos pode influênciar significativamente o desempenho final dos modelos. Vale ressaltar também que o Modelo 1 inclui notícias que se estendem até o ano de 2018, o que pode ter impactado os resultados na Figura 4.1, Figura 4.2 e Figura 4.3.

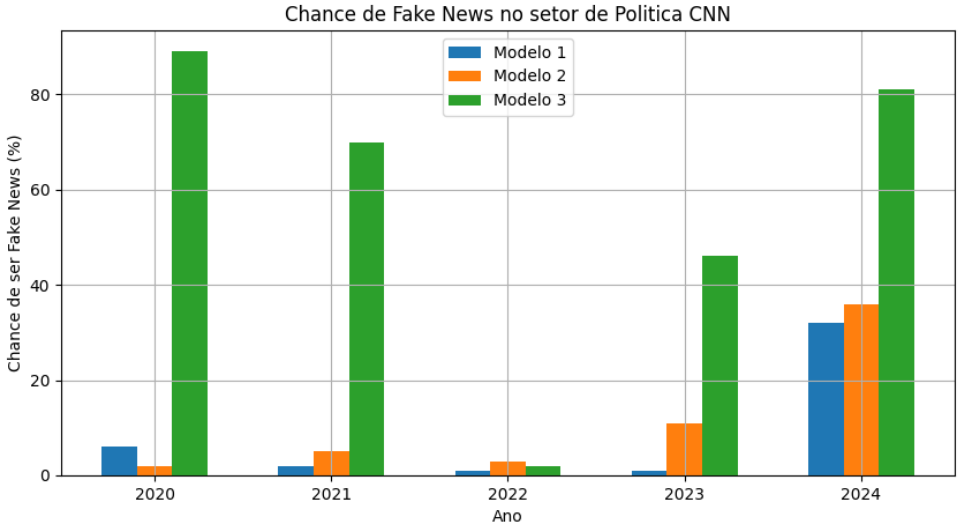

Figura 4.1 – Política CNN

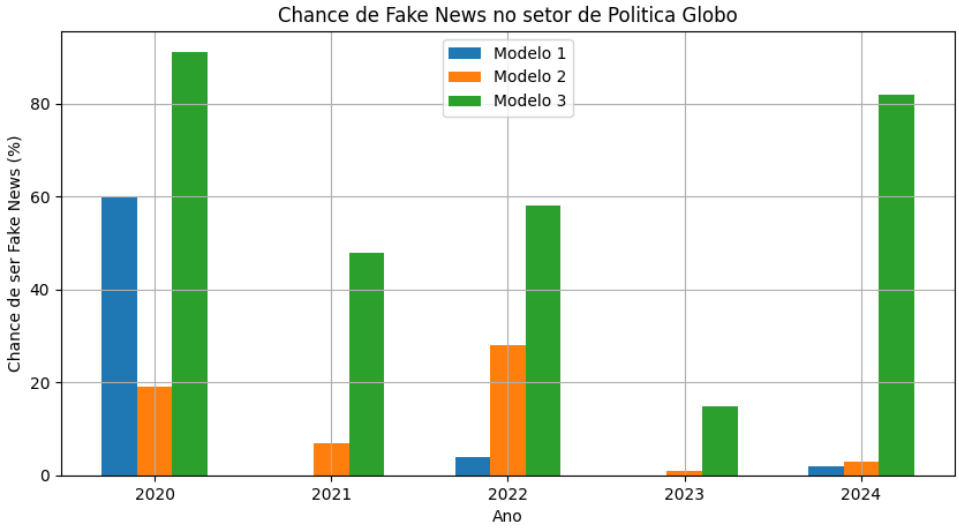

Figura 4.2 – Política Globo

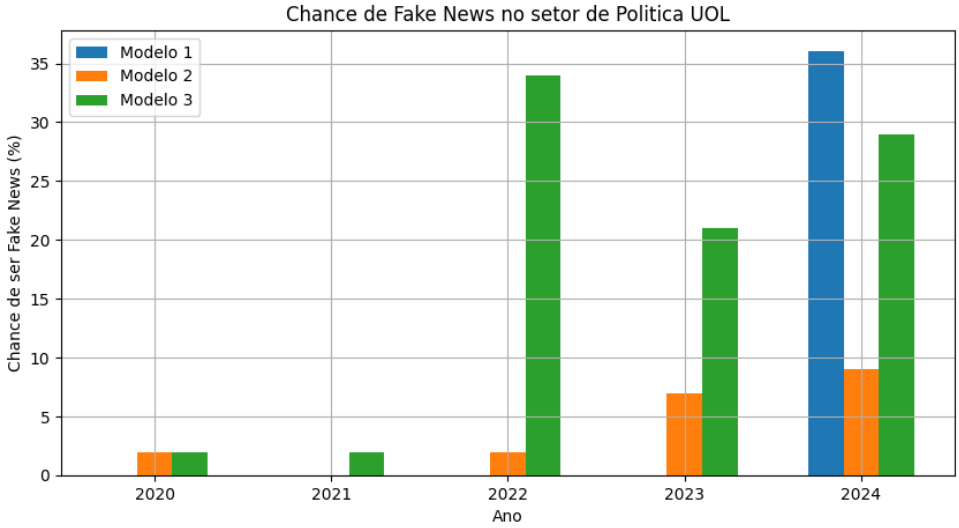

Figura 4.3 – Política UOL

4.2.2 Esportes

Considerando que essa categoria não foi amplamente representada em nenhum dos dois conjuntos de dados, com "TV & Celebridades" contendo 1.544 entradas e "Entretenimento" com 1.409 entradas, os resultados obtidos foram mais inconsistentes como mostra a Figura 4.4, Figura 4.5 e Figura 4.6.

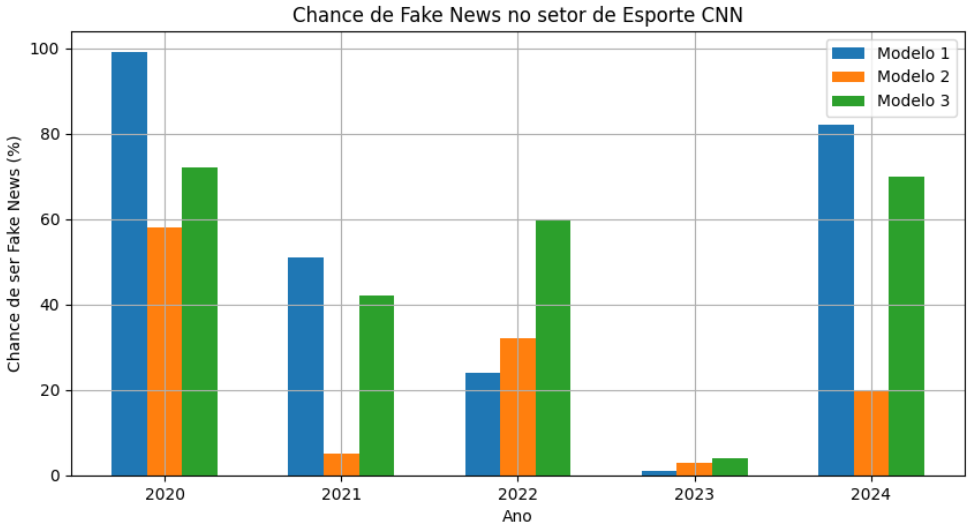

Figura 4.4 – Esportes CNN

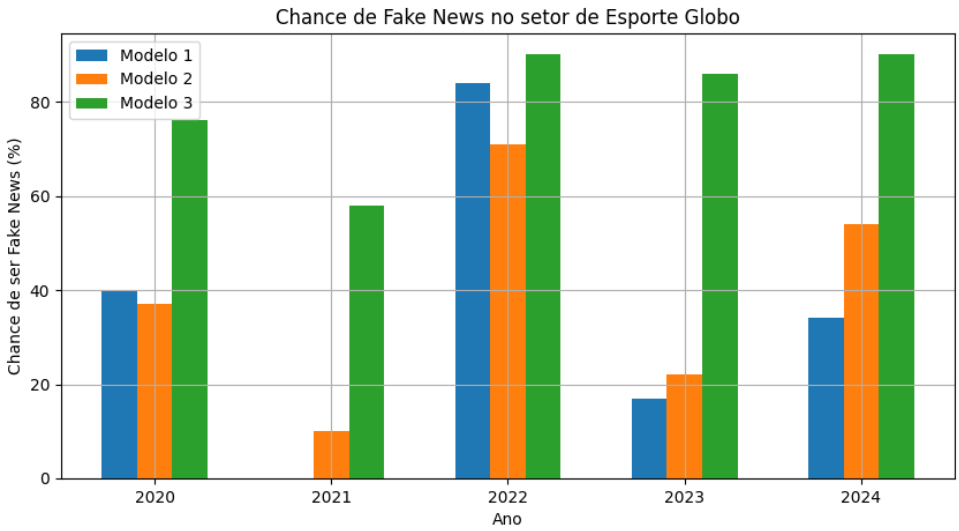

Figura 4.5 – Esportes Globo

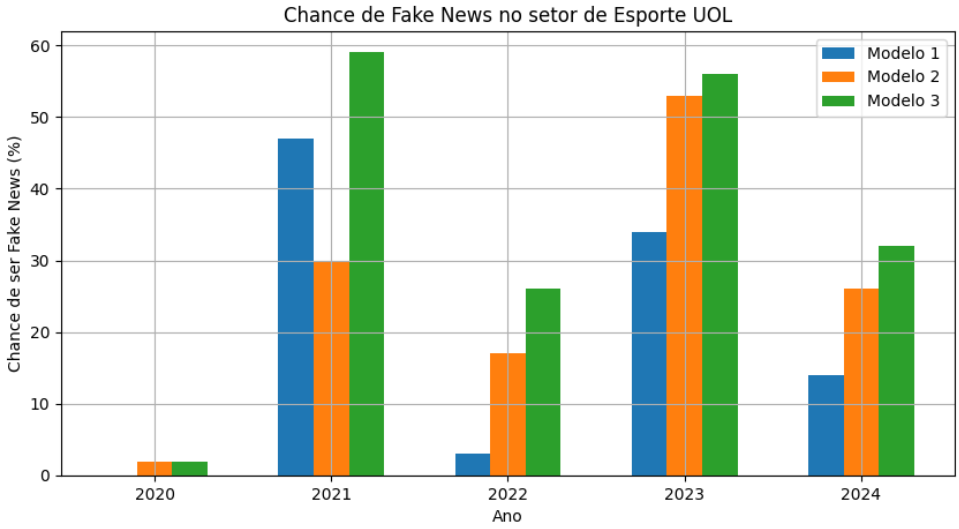

Figura 4.6 – Esportes UOL

Pode-se observar que algumas notícias não abordam diretamente o assunto, mas o utilizam como contexto para explicar outro tema. Um exemplo é a notícia de 2022 na categoria esporte da Globo, cujo título é "Quando a natureza inspira o esporte" (Gente, 2022).

Na Tabela 4.6 pode-se perceber que o assunto majoritário é a natureza, e a reportagem tenta estabelecer uma conexão desse tema com o esporte, o que pode ter ocasionado uma maior imprecisão no modelo.

Tabela 4.6 – Notícia Globo do ano 2022 na categoria "Esporte"

Que a natureza é fonte de inspiração, isso é inquestionável. A arte é uma prova disso. Mas você conseguiria associar um esporte com o futebol como algo que tem tudo a ver com a floresta? Basta pensar na quantidade de mascotes e símbolos compartilhados entre o futebol e as matas do Brasil. A equipe do Terra da Gente faz uma viagem do interior do estado de São Paulo até o norte do Brasil para mostrar quem são, na floresta, esses personagens que levam alegria e força aos times. Basta olhar para o apelido da Seleção Brasileira: “canarinho”, pelo canário-da-terra (Sicalis flaveola). Mas e o seu time do coração? Ele tem ligação com o meio ambiente? O mascote é um mamífero? Uma ave? Um anfíbio? Nossos aventureiros mostram que esporte e natureza têm uma ligação maior do que se imagina. E na mata, tem rivalidade? A equipe do repórter Paulo Augusto foi até ao rio Cuiabá para mostrar o dérbi do rio: um pontepretano e um bugrino pescam juntos numa aventura de muita competição sim, mas, principalmente, de muita emoção, respeito e amizade. O resultado desse jogo, ou melhor, dessa pescaria, você descobre na reportagem.

4.2.3 Economia

Considerando que o Modelo 1 possui apenas 44 entradas na categoria de Economia e que o Modelo 2 inclui dados políticos misturados com informações econômicas e é mais atualizado, de acordo com a Figura 4.7, Figura 4.8 e Figura 4.9 foi possível observar que o Modelo 2 apresentou melhores resultados e maior consistência. É importante destacar que, no site da CNN, onde as notícias são mais focadas nos aspectos técnicos da economia, o desempenho do Modelo 1 foi inferior. Em contraste, nos sites da Globo e UOL, que discutem como as decisões políticas influenciam a economia, os resultados do Modelo 1 se mostraram mais consistentes.

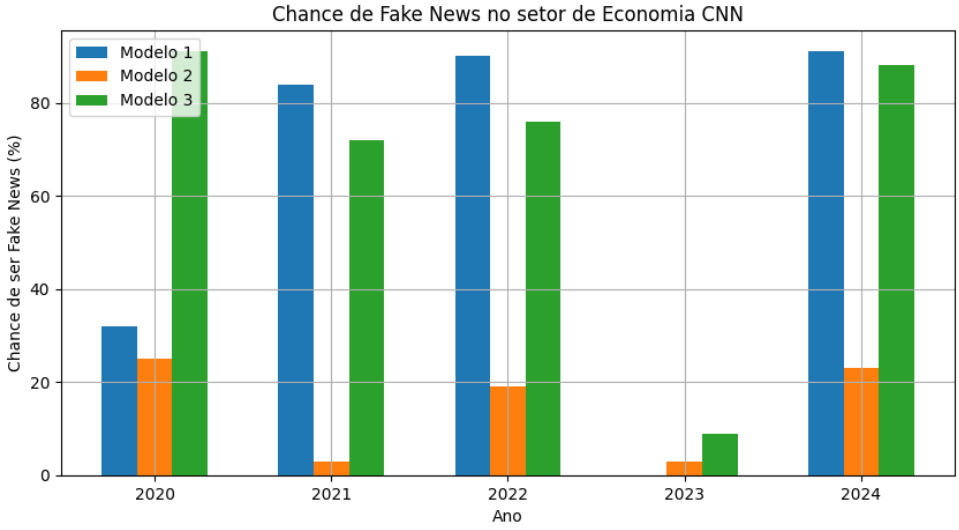

Figura 4.7 – Economia CNN

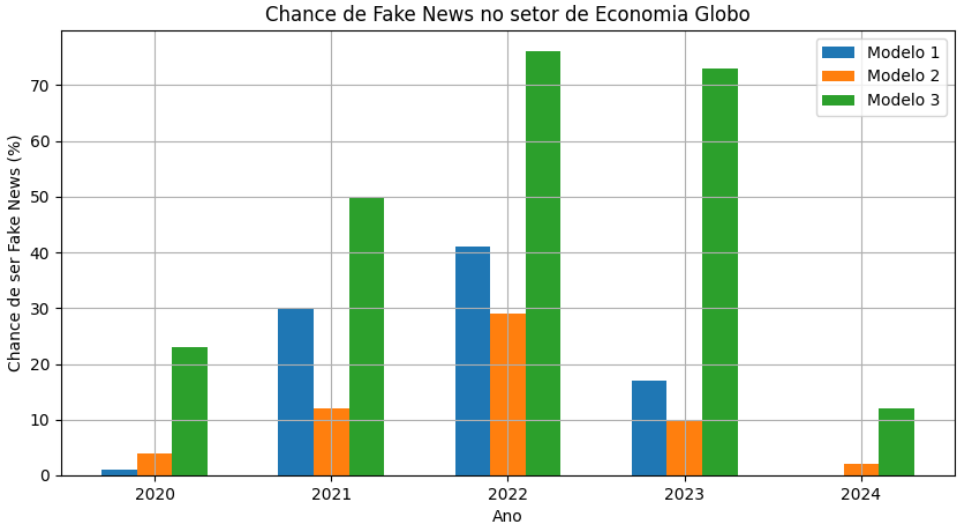

Figura 4.8 – Economia Globo

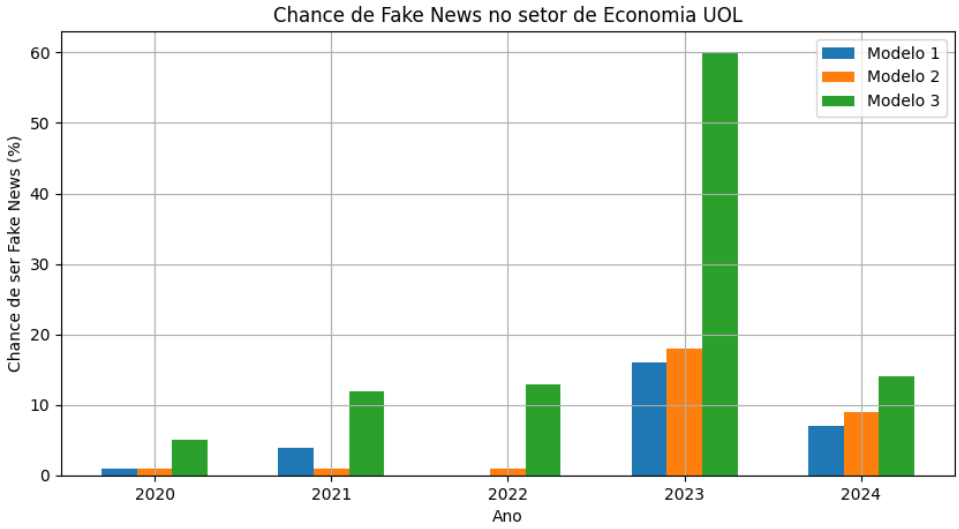

Figura 4.9 – Economia UOL

De acordo com a Tabela 4.7, temos o corpo da notícia de 2022 publicada pela CNN e pela Globo na categoria "Economia". É possível observar que a notícia da Globo utiliza menos termos técnicos relacionados à economia em comparação com a notícia da CNN, o que pode explicar a variação observada no Modelo 1.

Tabela 4.7 – Notícias do ano de 2022 na categoria "Economia"

CNN	Globo
Desempenho registrado neste ano é o terceiro melhor desde 2008, ficando atrás somente dos anos de 2016, 2015 e 2008. Em 2022, a Caderneta de Poupança registrou valorização de 7,9%, melhor desempenho desde 2016. Os dados foram levantados pela TradeMap e divulgados nesta sexta-feira (30). O desempenho registrado neste ano é o terceiro melhor desde 2008, ficando atrás somente dos anos de 2016 (8,30%), 2015 (8,15%) e 2008 (7,90%). O levantamento considera a rentabilidade da poupança desde o ano de 2000. A rentabilidade em 2022 deverá ter ganho real acima da inflação medida pelo IPCA. Uma prévia considerando a inflação até novembro de 2022 registra ganho de poder aquisitivo de 2,63%...	O encontro “Fortalecendo o Ecossistema de Bioeconomia” aconteceu na última quarta-feira (21), no Sidia Amazon Innovation, e reuniu representantes de 31 instituições de tecnologia para um debate sobre os grandes desafios e expectativas para o desenvolvimento da bioeconomia no Amazonas. O evento foi realizado pelo Instituto de Conservação e Desenvolvimento Sustentável da Amazônia (Idesam), por meio do Programa Prioritário de Bioeconomia (PPBio), e Suframa, com apoio do Sidia Instituto de Tecnologia e Manaus Tech Hub. O diretor de Inovação em Bioeconomia do Idesam e coordenador do PPBio, Carlos Koury, destaca o papel da Zona Franca para a promoção da bioeconomia.“O modelo de promoção da bioeconomia criado pela Zona Franca, incentivado via recursos de pesquisa, desenvolvimento e inovação, está embasado em um arranjo de instituições que são catalisadoras de novos negócios. Assim, o PPBio, aceleradoras e incubadoras têm um papel muito importante e precisam estar juntos”, afirma Koury...

4.2.4 Saúde

Nos resultados finais da categoria Saúde, como mostrado na Figura 4.10, Figura 4.11 e Figura 4.12, observou-se que o desempenho do Modelo 1 foi significativamente inferior ao do Modelo 2. O Modelo 2 possui um volume maior de conteúdo relacionado à saúde, com 4.456 entradas no conjunto de dados, o Modelo 1 possui apenas 112 entradas. Em alguns casos, como nas notícias da UOL, foi possível identificar que algumas delas podem estar relacionadas à categoria "Sociedade & Notícias Diárias" dentro do conjunto de dados do Modelo 1, o que pode ter contribuído para a maior precisão desse modelo em alguns casos.

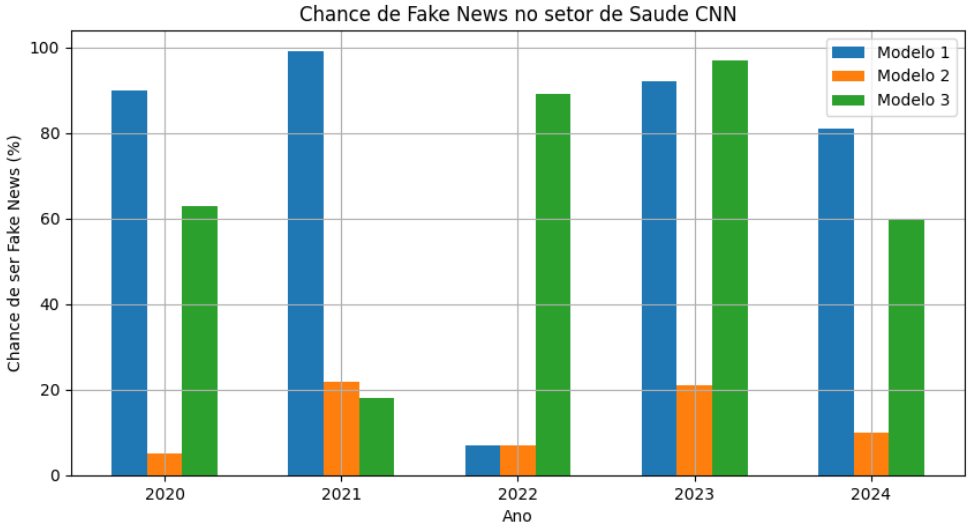

Figura 4.10 – Saúde CNN

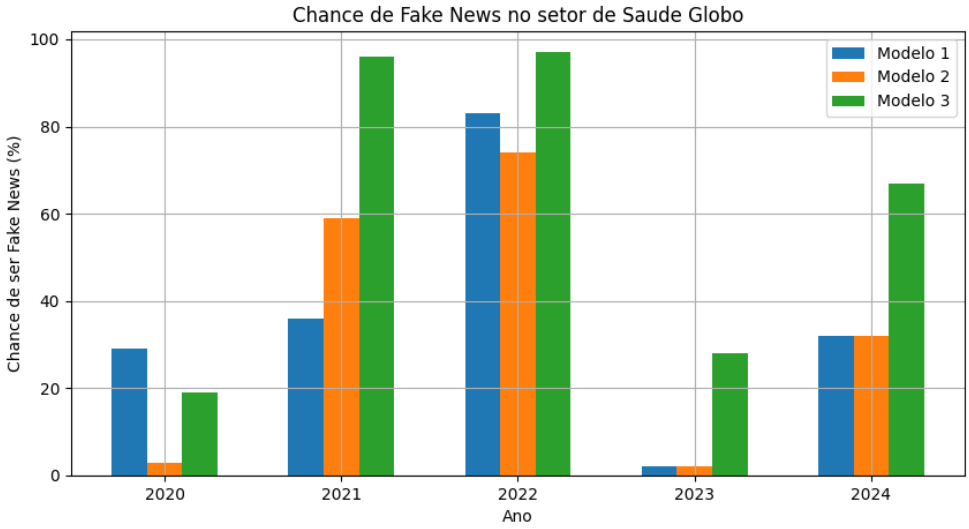

Figura 4.11 – Saúde Globo

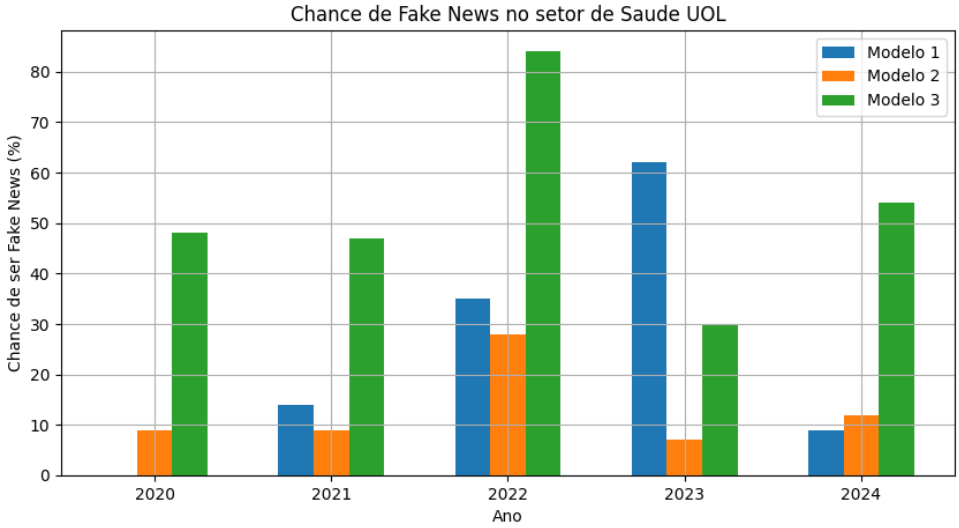

Figura 4.12 – Saúde UOL

Além dos Modelos 1 e 2, os resultados obtidos com o Modelo 3, que combina os conjuntos de dados do *FakeBR Corpus* e do *FakeRecogna*, foram menos satisfatórios. Esse resultado sugere que, embora o aumento na quantidade de entradas possa parecer vantajoso, o modelo acaba sendo treinado com base em duas estratégias distintas, o que pode comprometer sua eficácia. Além disso, o Modelo 3 inclui entradas do *FakeBR Corpus* que são válidas apenas até o ano de 2019, o que pode ter influenciado negativamente o desempenho geral.

Dessa forma, observou-se que o Modelo 2 apresentou maior acurácia nas categorias de Saúde e Economia, além de ter demonstrado bom desempenho na detecção de notícias falsas na categoria "Política". No entanto, o Modelo 1 foi superior nessa última categoria. Em contrapartida, nenhum dos modelos alcançou um desempenho satisfatório na categoria de Esportes.

É importante destacar que as notícias frequentemente abordam mais de um tema. Uma notícia sobre política, por exemplo, pode conter comentários sobre economia, enquanto uma notícia sobre saúde pode incluir referências a esportes. Acrescentando a isso, em um uso prático da aplicação, a acurácia pode variar não apenas entre as diferentes categorias, mas também entre as fontes de notícias, considerando que algumas podem tratar de determinados assuntos com mais profundidade do que outras.

Além disso, uma possível explicação para a grande discrepância nos resultados do modelo 3, em comparação com os modelos 1 e 2, pode estar na definição de notícias falsas dos conjuntos *FakeBR* e *FakeRecogna*. A ideia de combinar os dois conjuntos era obter maior precisão, porém, o resultado final foi um modelo que pode ter se tornado mais confuso, apresentando uma diferença significativa nos resultados para uma mesma notícia.

4.2.5 Uso da aplicação

No aspecto de usabilidade, observa-se que o acesso à ferramenta desenvolvida por (Silva; Fagundes, 2021) tornou-se significativamente mais simples. Anteriormente, era necessário treinar o modelo e executá-lo diretamente no código com informações específicas; agora, a ferramenta pode ser utilizada de forma dinâmica e acessível. Além disso, está disponível para um público sem conhecimentos em programação, democratizando seu uso. É possível analisar a usabilidade no vídeo publicado no YouTube[1], onde a Figura 4.13 representa um exemplo.

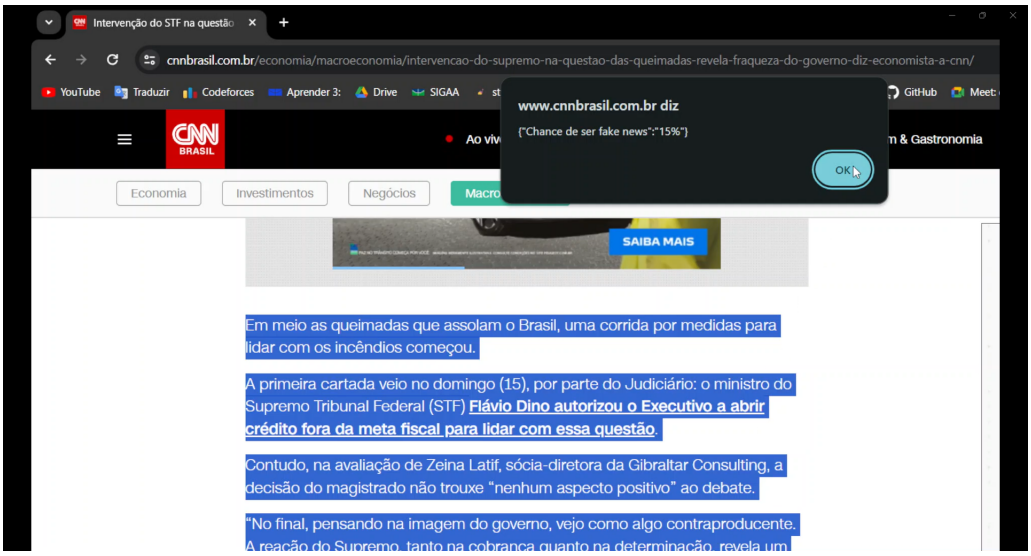

Figura 4.13 – Exemplo de uso da aplicação

Dessa forma, no contexto do *front-end*, observa-se que, ao selecionar o texto, é possível obter a resposta, porém apenas a notícia selecionada pode ser enviada para a API, o que pode comprometer tanto a usabilidade quanto o resultado final da análise. Outro ponto relevante é a seleção do modelo. Atualmente, a aplicação não permite que o usuário escolha qual modelo utilizar para verificar a veracidade da notícia, sendo o modelo padrão adotado o modelo 2, treinado exclusivamente com o *FakeRecogna*. Por fim, recomenda-se que o texto final seja ampliado, a fim de proporcionar maior clareza e visibilidade aos resultados.

No que se refere à API, o principal ponto crítico relacionado à usabilidade é a incerteza do usuário em relação a qual resultado confiar como final. Durante os testes, observou-se que os resultados obtidos variaram significativamente entre os modelos, de modo que, em um modelo, o resultado pode ser completamente distinto daquele gerado por outro. Essa discrepância pode gerar dúvidas no usuário final sobre qual resultado deve ser considerado confiável.

[1] https://youtu.be/F7C0bJSRbAs

5 Conclusões

A disseminação de notícias falsas representa um desafio complexo para a sociedade moderna, como discutido ao longo deste trabalho. Espalhar conhecimento verdadeiro e disponibilizar ferramentas para identificar a desinformação é essencial no combate à proliferação de notícias falsas. Durante o estudo, foi possível retreinar um modelo existente com um novo conjunto de dados e disponibilizar essa ferramenta para o público de forma prática com o desenvolvimento de uma extensão para o navegador Google Chrome. Também se analisou seu comportamento em notícias cotidianas, que frequentemente apresentam informações limitadas e temas variados.

Acredita-se que este estudo tenha contribuído para a comunidade ao avaliar a viabilidade e utilidade da extensão desenvolvida, demonstrando que seu uso faz sentido. Considerando que no trabalho anterior (Silva; Fagundes, 2021) o modelo estava indisponível para uso cotidiano, torná-lo acessível em uma aplicação backend consumida por uma extensão do Google Chrome facilita o acesso ao público geral. Com isso, o usuário pode selecionar o texto e clicar para consultar a resposta do modelo treinado, tornando o processo simples e prático. Conclui-se, assim, que a extensão desenvolvida para o Google Chrome representou um avanço significativo na facilitação do uso por parte de usuários comuns.

5.1 Trabalhos futuros

Para aumentar a acurácia do modelo atual, é recomendada, em um trabalho futuro, a criação de um banco de dados seguindo a mesma estratégia utilizada pelo *FakeBR Corpus*, porém com notícias mais recentes, similar ao que é feito no *FakeRecogna*. Tanto a estratégia de seleção aleatória de notícias falsas e verdadeiras utilizada pelo *FakeRecogna* quanto a limitação do tempo imposta pelo *Corpus Fake.BR* influenciam na qualidade dos modelos gerados. Além disso, definir um conjunto de dados utilizando a mesma definição de notícias falsas do *Corpus Fake.BR* pode ser uma possibilidade para reduzir a variância observada nos resultados do modelo 3 em comparação com os outros modelos.

Além disso, é relevante destacar que a usabilidade da extensão pode ser aprimorada, permitindo a seleção automática de toda a notícia, sem a necessidade de intervenção manual por parte do usuário final. Dessa forma, ele poderá obter uma análise geral da notícia de maneira mais simplificada. Ademais, uma possibilidade a ser considerada em trabalhos futuros é o aprimoramento do resultado apresentado ao usuário, podendo ser uma média dos resultados obtidos pelos três modelos, de modo a oferecer uma única fonte de resposta.

Também é recomendável utilizar um modelo mais recente, como o BERT (*Bidirectional Encoder Representations from Transformers*) (Devlin *et al.*, 2019). Esse modelo pode ser

ajustado para diversas tarefas de NLP, como resposta a perguntas e inferência de linguagem, sem a necessidade de modificações significativas em sua arquitetura. O BERT alcançou resultados de ponta em 11 tarefas de NLP.

Referências

ABDULRAHMAN, A.; BAYKARA, M. Fake news detection using machine learning and deep learning algorithms. *In*: **2020 International Conference on Advanced Science and Engineering (ICOASE)**. [*S.l.: s.n.*], 2020. p. 18–23. Citado na p. 14.

AHMAD, I.; YOUSAF, M.; YOUSAF, S.; AHMAD, M. O. Fake news detection using machine learning ensemble methods. **Complexity**, v. 2020, n. 1, p. 8885861, 2020. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1155/2020/8885861. Citado na p. 14.

ALPAYDIN, E. **Introduction to Machine Learning**. Fourth. [*S.l.*]: MIT Press, 2020. ISBN 978-0262043793. Citado nas pp. 17, 18 e 19.

BRASIL, C. **CNN Brasil - Notícias ao vivo, análises e informações**. 2024. Disponível em: https://www.cnnbrasil.com.br/. Citado na p. 26.

COPPERWAITE, C. L. M. **Learning Flask Framework**. Packt Publishing, 2015. Disponível em: https://books.google.com.br/books?id=HPGoCwAAQBAJ. Citado nas pp. 23, 25, 26 e 39.

DEBUGBEAR. **Counting Chrome Extensions – Chrome Web Store Statistics**. 2024. Disponível em: https://www.debugbear.com/blog/counting-chrome-extensions. Citado na p. 22.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *In*: **Proceedings of the 2019 Conference of the North**. Stroudsburg, PA, USA: Association for Computational Linguistics, 2019. v. 1, p. 4171–4186. ISBN 978-1-950737-13-0. ArXiv: 1810.04805. Disponível em: http://aclweb.org/anthology/N19-1423. Citado na p. 58.

FLETCHER, R.; CORNIA, A.; GRAVES, L.; NIELSEN, R. K. **Measuring the Reach of "Fake News"and Online Disinformation in Europe**. 2018. Disponível em: https://reutersinstitute.politics.ox.ac.uk/our-research/measuring-reach-fake-news-and-online-disinformation-europe. Citado na p. 11.

GARCIA, G. L.; AFONSO, L. C. S.; PAPA, J. a. P. Fakerecogna: A new brazilian corpus for fake news detection. *In*: **Computational Processing of the Portuguese Language: 15th International Conference, PROPOR 2022, Fortaleza, Brazil, March 21–23, 2022, Proceedings**. Berlin, Heidelberg: Springer-Verlag, 2022. p. 57–67. ISBN 978-3-030-98304-8. Disponível em: https://doi.org/10.1007/978-3-030-98305-5_6. Citado nas pp. 8, 12, 13, 26, 29 e 47.

GENTE, G. T. da. **Quando a natureza inspira o esporte**. 2022. Acessado em: 05 de setembro de 2024. Disponível em: https://g1.globo.com/sp/campinas-regiao/terra-da-gente/noticia/2022/08/19/quando-a-natureza-inspira-o-esporte.ghtml. Citado na p. 51.

GLOBO, G. O. P. de Notícias da. **G1 - O Portal de Notícias da Globo**. 2024. Disponível em: https://g1.globo.com/. Citado na p. 26.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [*S.l.*]: MIT Press, 2016. http://www.deeplearningbook.org. Citado nas pp. 15 e 17.

GUPTA, S.; PAL, R.; MISTRY, D.; SHARMA, A. Covid-19 infodemic: Impact on global health and nutrition. **Journal of Global Health**, National Center for Biotechnology Information (NCBI), v. 10, n. 1, p. 10093785, 2023. Disponível em: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10093785/. Citado na p. 11.

HARARI, Y. N. **Sapiens: A Brief History of Humankind**. 13ª. ed. New York, NY: Harper, 2015. ISBN 9780062316097. Citado na p. 19.

INSTAGRAM. **Combate à desinformação no Instagram**. 2022. Blog do Instagram. Acesso em: 20 jul. 2024. Disponível em: https://about.instagram.com/blog/announcements/combatting-misinformation-on-instagram. Citado na p. 14.

JOACHIMS, T. Text categorization with support vector machines: Learning with many relevant features. *In*: **European Conference on Machine Learning**. [*S.l.*]: Springer, 1998. p. 137–142. Citado na p. 19.

LAB, D. R. **Fact-Checking**. 2024. Acesso em: 5 setembro 2024. Disponível em: https://reporterslab.org/fact-checking/. Citado na p. 31.

LAI, S. Data misuse and disinformation: Technology and the 2022 elections. **Brookings Institution**, 2022. Disponível em: https://www.brookings.edu/articles/data-misuse-and-disinformation-technology-and-the-2022-elections/. Citado na p. 11.

LANGUAGES, O. **Dicionário de Português**. 2021. Disponível em: https://languages.oup.com/google-dictionary-pt/. Citado na p. 19.

MANNING, C. D.; SCHüTZE, H. **Foundations of Statistical Natural Language Processing**. Cambridge, MA: MIT Press, 1999. Citado nas pp. 19 e 20.

MEHTA, P. **Creating Google Chrome Extensions**. Apress, 2016. (Expert's voice in Web development). ISBN 9781484517758. Disponível em: https://books.google.com.br/books?id=diTUtAEACAAJ. Citado na p. 21.

META. **Informações sobre a empresa Meta**. 2024. About Meta. Acesso em: 20 jul. 2024. Disponível em: https://about.meta.com/br/company-info/. Citado na p. 14.

MITCHELL, T. M. **Machine Learning**. McGraw-Hill, 1997. Disponível em: https://www.cs.cmu.edu/~tom/mlbook.html. Citado na p. 15.

MONTEIRO, R. A.; SANTOS, R. L. S.; PARDO, T. A. S.; ALMEIDA, T. A. de; RUIZ, E. E. S.; VALE, O. A. Contributions to the study of fake news in portuguese: New corpus and automatic detection results. *In*: **Computational Processing of the Portuguese**

Language. [*S.l.*]: Springer International Publishing, 2018. p. 324–334. ISBN 978-3-319-99722-3. Citado nas pp. 12, 13, 14, 26 e 27.

MUREL, J.; KAVLAKOGLU, E. **Bag of Words (BoW) Explained**. 2024. Acesso em: 05 de setembro de 2024. Disponível em: https://www.ibm.com/topics/bag-of-words. Citado nas pp. 7, 32, 33 e 34.

MUSGROVE, A. T.; WILKINS, L.; ROGERS, K.; HAMMOND, R. Real or fake? resources for teaching college students how to identify fake news. **College & Undergraduate Libraries**, v. 25, n. 3, p. 243–260, 2018. Disponível em: https://doi.org/10.1080/10691316.2018.1480444. Citado na p. 14.

Olhar Digital. **Quais os navegadores mais usados do Brasil?** 2024. Acessado em: 07 de setembro de 2024. Disponível em: https://olhardigital.com.br/2024/02/19/internet-e-redes-sociais/quais-os-navegadores-mais-usados-do-brasil/. Citado na p. 21.

ONLINE, U. U. **UOL - O melhor conteúdo**. 2024. Disponível em: https://www.uol.com.br/. Citado na p. 26.

PROJECT, N. **Natural Language Toolkit**. 2024. Acessado em: 05 de setembro de 2024. Disponível em: https://www.nltk.org/. Citado na p. 37.

Python Packaging Authority. **Python Packaging User Guide: Key Projects - venv**. 2024. Disponível em: https://packaging.python.org/pt-br/latest/key_projects/#venv. Citado na p. 24.

Python Packaging Authority. **Python Packaging User Guide: Key Projects - venv**. 2024. Disponível em: https://packaging.python.org/pt-br/latest/guides/installing-using-pip-and-virtual-environments/. Citado na p. 24.

Python Packaging Authority. **Python Packaging User Guide: Key Projects - venv**. 2024. Disponível em: https://packaging.python.org/pt-br/latest/key_projects/#pip. Citado na p. 24.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. 3ª. ed. [*S.l.*]: Prentice Hall, 2010. Citado na p. 19.

SANTOS, V. Souza dos; SERRANO, A. L. M.; BECKMAN, M. K.; PRACIANO, B. J. G.; FILHO, G. P. R.; CANEDO, E. D. Utilização de inteligência artificial para verificar notícias falsas com a utilização do chatgpt. *In*: IADIS. **Proceedings of the CIAWI Conference 2023**. [*S.l.*], 2023. Citado na p. 12.

SCARABELLI, A. C. P. Fake news in the brazilian elections. **Diggit Magazine**, 2018. Disponível em: https://www.diggitmagazine.com/articles/fake-news-brazilian-elections. Citado na p. 11.

SHARMA, U.; SARAN, S.; PATIL, S. M. Fake news detection using machine learning algorithms. **Department of Information Technology, Bharati Vidyapeeth College of Engineering**, Navi Mumbai, India, 2020. Citado na p. 14.

SILVA, R. A. da; FAGUNDES, D. P. de C. **Estudo e Aplicação de SVMs na Detecção de Fake-News**. 2021. Citado nas pp. 12, 13, 26, 32, 38, 57 e 58.

SILVA, R. M.; SANTOS, R. L.; ALMEIDA, T. A.; PARDO, T. A. Towards automatically filtering fake news in portuguese. **Expert Systems with Applications**, v. 146, p. 113199, 2020. ISSN 0957-4174. Disponível em: http://www.sciencedirect.com/science/article/pii/S0957417420300257. Citado nas pp. 8, 27 e 46.

TRUELIST. **Google Chrome Statistics 2024**. 2024. Disponível em: https://www.truelist.co/blog/google-chrome-statistics. Citado na p. 22.

Union for International Cancer Control. **Fake Medical News and Other Costly Misinformation on Cancer**. 2021. https://www.uicc.org/news/fake-news-threat-public-health. Accessed: 2024-09-04. Citado na p. 11.

VAPNIK, V. N. Constructing learning algorithms. *In*: **The Nature of Statistical Learning Theory**. [*S.l.*]: Springer, 1995. p. 119–166. Citado na p. 18.

VAROQUAUX, G.; GRISEL, O.; CANDES, E. **Joblib: Lightweight Pipelines for Python**. 2024. Acessado: 2024-06-09. Disponível em: https://joblib.readthedocs.io/en/stable/. Citado nas pp. 24, 26 e 39.

VOSOUGHI, S.; ROY, D.; ARAL, S. The spread of true and false news online. **Science**, American Association for the Advancement of Science, v. 359, n. 6380, p. 1146–1151, 2018. Disponível em: https://science.org/doi/10.1126/science.aap9559. Citado na p. 11.