UNIVERSIDADE DO ESTADO DE SANTA CATARINA – UDESC
CENTRO DE EDUCAÇÃO SUPERIOR DO ALTO VALE DO ITAJAÍ – CEAVI
ENGENHARIA DE SOFTWARE

ANDREW VINICIUS DA SILVA BAASCH

APLICAÇÃO DE PROCESSAMENTO DE LINGUAGEM NATURAL PARA ANÁLISE DE TEXTO E INDICAÇÃO DE NOTÍCIAS SIMILARES: UMA FERRAMENTA DE APOIO PARA A IDENTIFICAÇÃO DE *FAKE NEWS*

IBIRAMA

2021

ANDREW VINICIUS DA SILVA BAASCH

APLICAÇÃO DE PROCESSAMENTO DE LINGUAGEM NATURAL PARA ANÁLISE DE TEXTO E INDICAÇÃO DE NOTÍCIAS SIMILARES: UMA FERRAMENTA DE APOIO PARA A IDENTIFICAÇÃO DE *FAKE NEWS*

Trabalho de conclusão apresentado ao curso de Engenharia de Software do Centro de Educação Superior do Alto Vale do Itajaí (CEAVI), da Universidade do Estado de Santa Catarina (UDESC), como requisito parcial para a obtenção do grau de bacharel em Engenharia de Software.

Orientadora: Marília Guterres Ferreira

IBIRAMA

2021

ANDREW VINICIUS DA SILVA BAASCH

APLICAÇÃO DE PROCESSAMENTO DE LINGUAGEM NATURAL PARA ANÁLISE DE TEXTO E INDICAÇÃO DE NOTÍCIAS SIMILARES: UMA FERRAMENTA DE APOIO PARA A IDENTIFICAÇÃO DE *FAKE NEWS*

Trabalho de conclusão apresentado ao curso de Engenharia de Software do Centro de Educação Superior do Alto Vale do Itajaí (CEAVI), da Universidade do Estado de Santa Catarina (UDESC), como requisito parcial para a obtenção do grau de bacharel em Engenharia de Software.

Banca Examinadora

Orientador:

Profª. Drª. Marília Guterres Ferreira
UDESC

Membros:

Profª. Drª. Lívia Ruback
Universidade Federal Rural do Rio de Janeiro (UFRRJ)

Prof. M.Sc. Mattheus da Hora França
UDESC

Ibirama, XX/XX/2021

AGRADECIMENTOS

Inicialmente, gostaria de agradecer minha mãe, Beatriz Aparecida da Silva, por ter me auxiliado e não ter me deixado desistir.

Agradeço minha amiga Silvana Nascimento, por dedicado um tempo para revisar algumas partes do meu trabalho.

Agradeço muito a meu amigo Rodrigo Valle, por ter me dado algumas dicas, por ter me auxiliado no teste do meu sistema, muito obrigado mesmo.

Agradeço muito minha orientadora Profª. Drª. Marília Guterres Ferreira, por ter me incentivado, me auxiliado, e não ter desistido de mim, meu muito obrigado.

Por fim gostaria de agradecer a todos que ajudaram a chegar até aqui, direta ou indiretamente, meus mais profundos agradecimentos.

RESUMO

Identificar a origem de alguma notícia, pode ser difícil, principalmente com a imensa quantidade de informações que existe hoje. Procurando auxiliar o usuário nesta missão, este trabalho desenvolveu um sistema com a capacidade de adquirir, e posteriormente apresentar ao usuário notícias similares à aquela previamente fornecida ao sistema, utilizando Processamento de Linguagem Natural, disponibilizando informação para que o usuário possa realizar uma análise mais criteriosa.

Palavras-chave: Informação similar. Processamento de Linguagem Natural. *Fake News*.

ABSTRACT

Identifying the source of some news can be difficult, especially with the massive amount of information that exists today. Seeking to assist the user in this mission, this work developed a system with the ability to acquire, and subsequently present to the user similar news to that previously provided to the system, using Natural Language Processing, providing information so that the user can perform a more careful analysis .

Keywords: Similar information. Natural Language Processing. Fake News.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BBC	*British Broadcasting Corporation*
POC	Prova de Conceito
PLN	Processamento de Linguagem Natural
OMS	Organização Mundial da Saúde
NLTK	*Natural Language Toolkit*
TF-IDF	Termo de Frequência - Inverso da Frequência nos Documentos
API	Interface de Programação de Aplicações
TCC	Trabalho de Conclusão de Curso

SUMÁRIO

1 INTRODUÇÃO

Com o surgimento da internet e eventual criação e disseminação das redes sociais, surgiu um ambiente onde tornou-se possível compartilhar ideias e opiniões, que tinham o potencial de acabar gerando discussões acerca dos assuntos expostos, gerando engajamento, interação e participação dos usuários. No entanto, logo foi percebido o potencial nocivo que estes ambientes acarretavam, por exemplo, publicações realizadas por usuários, muitas vezes sem a checagem de fontes, tendendo a dispor de um conteúdo minimamente duvidoso. Faustino (2020) afirma que estas publicações são limitadas muitas vezes apenas pela rede de contatos do usuário, estas acabando por ocupar o mesmo patamar de relevância de conteúdos postados por grandes meios de comunicação, que em tese precisam manter a credibilidade das informações que publicam.

Neste contexto, como averiguou BBC (2020), as *fake news* se tornaram mais evidentes, espalhando-se de forma exponencial, disseminando desinformação e causando um pânico desnecessário.

Segundo Gelfert (2018), *fake news* se traduz como a disseminação de conteúdo propositalmente falso ou enganoso em um formato que se assemelha a uma notícia verídica. Embora o termo venha sendo bastante empregado ultimamente ele não é novo. Como mostra Merriam-Webster (2020), o uso geral do termo começou por volta do fim do século XIX. O termo ganhou mais destaque na disputa presidencial estadunidense no ano de 2016, entre os candidatos Hillary Clinton e Donald Trump, onde as *fake news* foram cogitadas como um dos motivos do resultado da eleição naquele ano, como mostra BBC (2018).

Antes da popularização da internet a disseminação de notícias falsas era algo muito mais trabalhoso, pois enfrentava alguns obstáculos como: custo alto, falta de flexibilidade, falta de conhecimento sobre o leitor e ausência de contexto. Porém, hoje essas barreiras foram superadas, tornando acessível a qualquer um criar e distribuir falsas notícias, como mostra Itagiba (2017). Além disso, a popularização da internet também proporcionou que ela se tornasse um ambiente de manifestação de ideias e individualidade, que permitiu que todos tenham a possibilidade de se tornar um divulgador de notícias, com a capacidade de alcançar um grande número de pessoas, cuja quantidade depende da extensão da rede de conexões do divulgador de conteúdo (FAUSTINO, 2020).

Fake news geralmente são criadas com o propósito de espalhar desinformação. Conforme Dicionário Priberam da Língua Portuguesa (2020), sua definição é: "Ato ou efeito de desinformar, de suprimir uma informação, de minimizar a sua importância ou de modificar o seu sentido". Além disso, *fake news* também são lançadas para gerar alarde e medo, a fim de criar uma comoção desnecessária. Também podem ser feitas com o objetivo de ganhar dinheiro,

como demonstrado no relato (BBC, 2019), em que é possível visualizar como existem pessoas dispostas a se organizarem a fim de obter vantagem financeira sobre outras pessoas.

O fato de qualquer informação estar a um clique de distância de modo quase instantâneo, fez com que ficássemos com uma sede por novidades, uma forte característica da era da informação. No entanto isso veio com grande lado negativo, grande parte da população parou de se preocupar com a fonte da informação que recebe, estando mais preocupada com a velocidade em que se recebe alguma informação, como mostra (FAUSTINO, 2020). Ainda segundo Faustino (2020), é possível perceber uma grande relação das *fake news* com as redes sociais, onde surgiu uma necessidade de ser o primeiro a divulgar ou falar sobre um fato ou acontecimento a fim de ganhar visibilidade e curtidas, não se importando com a origem ou a fonte daquela informação.

A grande aceitação das tecnologias de comunicação, por uma parcela relevante da população, ocasionou uma explosão de informação, onde qualquer pessoa com acesso à internet se torna um potencial divulgador de conteúdo (PRIMO, 2011). Essa massiva quantidade de informação ocasionou o que veio a ser chamado de Infodemia, que se refere a epidemia de informação, que acomete o mundo atualmente (THE WASHINGTON POST, 2003).

Com o grande volume de informações disponíveis, se tornou simples selecionar a dedo os dados que auxiliem na criação de uma narrativa que distorce a realidade, fazendo com que a indignação, seja substituída pela indiferença, que leva a conveniência (JUNIOR, 2019). isso veio a ser chamado de Pós Verdade.

Algo que potencializou o impacto das *fake news* foram as Bolhas dos Filtros, que segundo Pariser (2012), "são mecanismos de previsão que criam e refinam constantemente uma teoria sobre quem somos e sobre o que vamos fazer ou desejar a seguir". Como mostra Sastre, Oliveira e Belda (2018), apesar de não ser o fator determinante, as bolhas dos filtros proporcionam um ambiente propício para a disseminação de notícias de conteúdo pouco relevante de fontes não confiáveis ou ausentes. O impacto que as Bolhas dos Filtros geram é auxiliado pelo *Viés de Confirmação*, que é a tendência de se procurar fatos que reforcem crenças ou ideias já pré-concebidas, o que torna o usuário muito suscetível as Bolhas dos Filtros, pois elas apresentam conteúdos com os quais ele já concorda.

Com a massiva proliferação das *fake news*, começaram a surgir agências de checagem de fatos ou *fact-checking*. Segundo Fatos (2021b), "A checagem de fatos é um método jornalístico por meio do qual é possível certificar se a informação apurada foi obtida por meio de fontes confiáveis e, então, avaliar se é verdadeira ou falsa, se é sustentável ou não".

Assim para fornecer apoio na identificação de *fake news*, neste trabalho, foi desenvolvido um bot inteiramente, que será disponibilizado para uso de usuários comuns. Para isso, o mesmo será disponibilizado na internet. Para melhorar a usabilidade e interação com o público, o bot

foi desenvolvido para interagir através de um aplicativo de mensagens Telegram. O bot será nomeado de APLNATINS, o código do bot pode ser visualizado no meu repositório do GitHub: https://github.com/Volpe6/TelegramBotPython.

A interação do usuário é feita unicamente por meio de texto, o bot apresenta mensagens que indicam a etapa atualmente em execução. Ao final do processamento realizado o sistema apresenta ao usuário quais foram as notícias que obtiveram a maior similaridade com aquela apresentada por ele.

Os locais nos quais o bot pesquisa as notícias similares são definidas previamente. Na primeira versão do sistema o bot sempre irá realizar buscas nestes locais já definidos não podendo ser alterado pelo usuário.

1.1 PROBLEMA

No geral, ao se depararem com uma nova informação, uma relevante parcela das pessoas não sente a necessidade de verificar se determinada informação possui alguma veracidade, por exemplo, indo atrás de sua fonte. Apenas se pressupõem que se trata de algo verídico pelo fato de ser um familiar ou amigo quem compartilhou. Outro fator que geralmente faz com que as pessoas acreditem na notícia é o fato de algumas citarem nomes de entidades sérias, o que acaba por fornecer "credibilidade"à notícia (PSYCHOLOGY TODAY, 2018).

A disseminação de *fake news*, pode vir a acarretar diversas adversidades, dentre elas a mais grave envolve saúde pública. Por conta do alastramento das *fake news*, começou a surgir um movimento antivacinação. Suas teses e seus argumentos se baseiam em informações falsas, ou inventadas sendo propagadas por charlatões, que muitas vezes querem apenas promover seus produtos, mesmo que seja ao custo de vidas humanas (SARAIVA; FARIA, 2019). Outro grande problema ocasionado pelas *fake news* é violência, após a circulação de boatos sem nenhuma comprovação uma pessoa acabou morta (G1, 2017). Acusações falsas que mancham a imagem de indivíduos, como o que ocorreu com a canditada a presidência estadunidense Hillary Clinton em 2016 (BBC, 2018).

No atual momento de pandemia no qual o mundo se encontra, Trotta (2020) destaca que a disseminação de *fake news*, em especial na área da saúde, aumentou, seja pela indicação de medicamentos milagrosos ou pela indicação de terapias alternativas, propagando desinformação e afetando a crença da população em medidas realmente eficazes.

Usuários tendem a dar muita credibilidade a informações que circulam nas redes sociais. Este alto nível de confiança faz com que seja menos provável que verifiquem a origem de uma notícia informada por estes meios, isso os torna mais suscetíveis a compartilharem notícias sem verificar sua procedência (TALWAR et al., 2019).

Algo que também auxilia na criação de *fake news* é o anonimato que a rede pode proporcionar, pois o anonimato passa uma falsa sensação de impunidade por aqueles que criam e divulgam este tipo de informação enganosa (FAUSTINO, 2020).

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Este trabalho procura fornecer uma ferramenta acessível, que possa fornecer a seu utilizador um ponto de partida, por meio de, ao informar uma notícia duvidosa, o sistema retornará notícias similares, pelas quais o usuário poderá tirar a sua conclusão sobre a veracidade da notícia, ou aprofundar sua pesquisa. Seu objetivo geral é identificar notícias similares com aquela informada pelo usuário.

1.2.2 Objetivos Específicos

- Dado uma notícia, identificar a sentença mais importante.
- Identificar as palavras que melhor representa a sentença.
- A partir de uma base de busca para as notícias, recuperar aquelas com a maior semelhança com a notícia informada.
- Disponibilizar uma ferramenta web, para acesso de usuários comuns.

1.3 JUSTIFICATIVA

Como atualmente grande parte das notícias veiculadas são acessadas por meio das mídias sociais, a maioria dos usuários passou a depositar um alto nível de confiança nelas. Boa parte deles se encontram dentro de grupos onde aqueles que participam geralmente são de sua confiança. Desta forma o usuário passa a dar mais credibilidade às notícias dentro destes grupos, passa a compartilhá-las mais e cada vez menos busca checar as fontes (TALWAR et al., 2019).

1.4 METODOLOGIA

A análise do sistema foi feita em cima dos trabalhos correlatos. Como a especificação de requisitos, requisitos não funcionais, e regras de negócio.

Durante a fase do projeto para atender a os requisitos, foi realizada uma pesquisa buscando possíveis ferramentas para a utilização durante a implementação. Estas ferramentas deveriam fornecer um modo de interação com o usuário, e possibilitar a interação via código com a internet. Para a escolha destas ferramentas foram elencados dois critérios, facilidade no aprendizado, e ser desenvolvida na linguagem de programação Python. Para cada ferramenta utilizada foram realizadas provas de conceito (POC), a fim de atestar sua aplicabilidade no projeto.

Na parte responsável pela execução do tratamento de texto, foram utilizados ferramentas e algoritmos que tratam de Processamento de Linguagem Natural (PLN), para extração de informação, e tratamento de dados, antes de serem repassados ao usuário. Todo o sistema foi desenvolvido utilizando a linguagem de programação Python.

Foram realizados testes do nível de sistema, onde foi testado o projeto como um todo, verificando se as funcionalidades desenvolvidas estavam de acordo com os requisitos.

Para o controle das etapas a serem desenvolvidas, foram levantados os pontos essenciais para a implementação. Cada ponto posteriormente foi decomposto em pontos menores, sendo executados em sequência.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção são apresentados os conceitos tratados neste projeto. Aqui são debatidas as ferramentas que foram utilizadas para o desenvolvimento deste sistema, e os fundamentos que permitiram sua elaboração.

2.1 INFODEMIA

Termo cunhado em 2003 pelo jornalista David J. Rothkopf, que se refere a uma epidemia de informação, uma quantidade excessivamente grande e sem fundamento, que circula pelas tecnologias de comunicação, que prejudica a realização imediata e efetiva de ações envolvendo saúde pública, por exemplo. A infodemia é uma das enfermidades mais virulentas que o ser humano já presenciou, possui a capacidade de transitar por entre os continentes de modo quase instantâneo. Esta epidemia tem o potencial de desencadear comportamentos irracionais, desviar nossa atenção de problemas realmente importantes, dificultar o trabalho de profissionais e desestabilizar democracias (THE WASHINGTON POST, 2003).

A característica mais visível da infodemia é o excesso de informação, que facilita interpretações equivocadas que tende a distorcer a intencionalidade original de uma informação. Aliado a isso, esta a era atual na qual se vive, onde o indivíduo está o tempo todo conectado na rede (ALMEIDA, 2020).

Segundo a OMS (2018) infodemia pode ser definida como: rumores, fofocas, e informações que espalham-se internacional e instantaneamente, por meio do uso indiscriminado de celulares, mídia social, e demais tecnologias de comunicação.

Os efeitos da infodemia tentem a persistir, pois terá ocasionado problemas políticos, sociais, econômicos e de saúde pública. Embora seja possível controlar a infodemia, isso dependerá de fatores sociais, culturais e comportamentais da população, sendo necessário auxílio do estado com investimento em educação e campanhas de conscientização para toda a população (ARROYO-SÁNCHEZ; CABREJO; CRUZADO, 2020).

2.2 PÓS VERDADE

Segundo o Dicionário de Oxford (2021) consiste em, "Adj. Relacionado a circunstâncias em que as pessoas respondem mais a sentimentos e crenças do que a fatos". O que foi chamado de "pós verdade"ganhou relevância e atenção em 2016, quando o dicionário de Oxford nomeou esta como a palavra daquele mesmo ano. Após uma série de eventos e acontecimentos

a "verdade", aquela sustentada pelos fatos irrefutáveis, começou a perder espaço e relevância ao mesmo passo que notícias falsas e sem fundamento começavam a ganhar cada vez mais força (MCINTYRE, 2018). O termo pós verdade se consolidou em virtude do atual momento de polarização política que foi viabilizado por uma onda de negacionismo científico que foi reforçada pelas redes sociais, bolha dos filtros e *big data* (JUNIOR, 2019).

O fenômeno da pós verdade é caracterizado por uma negação sistêmica da realidade, sendo esta substituída por algo mais conveniente mesmo sendo completamente inventada e sem nocão. Isto só foi possível pois a verdade perdeu relevância não passando de um mero detalhe que alguns optam por ignorar (JUNIOR, 2019).

"Nesta era de política pós-verdade, é fácil selecionar os dados a dedo e chegar a qualquer conclusão desejada."(DICIONÁRIO DE OXFORD, 2021).

2.3 BOLHA DOS FILTROS

Em 2010, Como mostrou Pariser (2012), ocorreu aquilo que viria a ser talvez uma das maiores mudanças ocorridas na internet. Por meio de um anúncio sutil o Google anunciou que estaria disponibilizando a pesquisa personalizada para todos (GOOGLE BLOG, 2010). A princípio, essa parece ser uma mudança sem muito impacto, mas após isso a internet se transformaria permanentemente. Começava a era da personalização, onde o foco da rede tornou-se o usuário. A rede passou a refletir os interesses daquele que a utiliza, traçando um perfil do usuário. Hoje já passados 10 anos, não existe mais apenas um Google. Cada usuário que o utiliza tem uma experiência única, duas pessoas distintas a buscar um mesmo termo no Google, utilizando exatamente das mesmas palavras possuem a possibilidade de obter resultados bem variados.

Como menciona Pariser (2012), no início os entusiastas da internet acreditavam que ela seria descentralizadora e disruptiva ao tirar o controle de alguns poucos e redistribuir entre seus usuários, porém ainda como mostra Pariser (2012), a rede o está concentrando. Hoje o controle da internet se encontra centralizado na mão de poucas empresas, que praticamente ditam o rumo que ela deve tomar.

A internet tornou-se um local em que para se fazer quase qualquer coisa é necessário ceder algum tipo de informação. A rede alterou sua arquitetura, modificando-a de modo a induzir o usuário a ceder cada vez mais informação. O principal meio que a rede encontrou para induzir o usuário a compartilhar informações foi oferecer serviços gratuitos, onde muitos pedem que seja informado o nome, sobrenome, data de nascimento, endereço, entre outros. Isso faz com que a privacidade que existia antes seja trocada por um lugar mais cômodo. Segundo Pariser (2012), o motivo pelo qual a rede se modificou afim de obter tantos dados sobre os usuários,

foi devido ao fato das grandes companhias que detém um monopólio sobre ela, obter grande parte de sua renda provinda da publicidade que geram, e com a grande quantidade de dados que estas empresas possuem sobre os usuários, elas traçam um perfil altamente refinado, que as permitem, por exemplo, direcionar publicidade altamente personalizada, o que aumenta muito a probabilidade de surtir efeito no usuário.

A personalização afeta tudo que se é feito na internet. Como cita Pariser (2012), um exemplo disso são os *feeds* de notícias presente nas redes sociais, eles mostram cada vez mais conteúdos que refletem o interesse do usuário, fazendo-o a acreditar que todos ao seu redor compartilham das mesmas opiniões e crenças. Parafraseando Pariser (2012), este ambiente onde o conteúdo que se tem acesso apenas reforça opiniões e crenças, pode ser muito prejudicial, pois perde-se o contato com a realidade e fica cada vez mais difícil diferenciar o que é real daquilo que foi inventado.

Assim chega-se à bolha dos filtros. As bolhas são micro mundos criados sob medida para o usuário. São concebidas de modo que todo o conteúdo e informação que chega a ele reflita seus interesses (PARISER, 2012). Isso faz com que a internet se torne um local muito atraente. O grande problema nisso é que a grande maioria dos parâmetros que são utilizados para criar as bolhas não são conhecidos, isso é um grande limitador que torna muito difícil se saber qual tipo de conteúdo e informação está sendo deixado de fora. "Por não escolhermos os critérios que os sites usarão para filtrar os diversos assuntos, é fácil intuirmos que as informações que nos chegam através de uma bolha de filtros sejam imparciais, objetivas, verdadeiras. Mas não são."(PARISER, 2012). Ainda o grande motivo pelo qual os filtros foram tão bem aceitos, está no fato de termos que lidar com uma quantidade extremamente grande de informação, é humanamente impossível analisar toda a informação que se encontra disponível na rede. Isso fez com que os filtros se tornassem muito requisitados. Outro grande motivo que fez os filtros serem tão bem aceitos, está no fato de que aquilo que se procura na internet não ser necessariamente informações novas, mas sim informações que reforcem aquilo que já se sabe.

No mundo pré-internet, o principal meio de contato com os eventos que ocorriam ao redor do globo eram os jornais. Neles era onde se recolhiam informações acerca dos principais acontecimentos. Porém, as informações apresentadas pelos jornais são apenas a interpretação deles sobre os fatos que desejam transmitir. Os jornais eram os intermediários, eles que definiam o que era relevante e o que não era. Com a vinda de internet isso mudou, agora é possível checar as fontes diretamente, não estando mais preso a um único meio de comunicação. Ainda assim, em certo nível o intermediário ainda existe, a internet o tornou menos perceptível. Para verificar como intermediário ainda existe, basta verificar qual o meio mais utilizado para se obter informações hoje: quando se quer obter uma informação utiliza-se de motores de busca, para interagir com outros usuários da rede utiliza-se das redes sociais, e são os algoritmos

desenvolvidos pelas empresas que regem estes sistemas que determinam o que é visualizado e aquilo que possui maior relevância. A grande mudança que a internet trouxe na mídia fez com que o modo como se interage com a informação mudasse. Antes apenas um meio de comunicação unilateral disponibilizava notícias, hoje existem diversas outras fontes com as quais se é possível interagir, não sendo mais de um modo unilateral. Uma mudança significativa ocasionada pela internet foi a perda de confiança que os meios tradicionais de comunicação tiveram. Um dos motivos para que isso acontecesse foi porque ao se ter mais de uma fonte de informação, uma única fonte tende a perder confiança, por existirem várias interpretações de um mesmo fato (PARISER, 2012). Isso trouxe benefícios, não é mais necessário ficar atrelado a uma única fonte, isso possibilita atestar um fato ocorrido de pontos de vista distintos, o lado ruim, foi que uma notícia postada sem nenhum tipo de critério ou curadoria ganhou o mesmo espaço e destaque daquelas que tem (PARISER, 2012).

2.4 *FAKE NEWS*

Como mostra Ferreira (2018), existem alguns tipos de *fake news*, estas podem ser produzidas com os mais variados propósitos, dos mais inofensivos como uma sátira ou paródia, até os mais escusos, com intenção de difamar ou destruir a imagem de pessoas ou organizações baseado em conteúdo falso ou enganoso.

Com a grande popularização das redes sociais, elas se tornaram o meio pelo qual grande parte dos usuários da internet de alguns países acessa notícias, como mostra Fletcher e Nielsen (2017). No entanto as redes sociais foram concebidas de modo a fazer o usuário a despender grandes quantidades de tempo nelas, isso faz com que as notícias veiculadas não sejam as mais relevantes, mas sim aquelas que o usuário demonstra maior interesse. Deste modo, as informações acessadas contêm conteúdo sucinto e duvidoso, mas refletem os interesses do usuário, o que torna mais fácil aceitar a informação. Com isso as redes sociais se tornam um ambiente propício para disseminação de notícias falsas.

Aplicativos utilizados para interagir com outras pessoas como *WhatsApp*, também tendem a criar um terreno fértil para *fake news* (RESENDE et al., 2019). Como menciona Resende et al. (2019), as notícias falsas que circulam por meio do aplicativo tendem a possuir menos texto e apelar para o lado sentimental dos usuários, o que pode fazer com sintam a necessidade de compartilhá-las.

Segundo SANTOS e TEIXEIRA (2019), *fake news* podem ser caracterizadas como um modo de desinformação, um meio de gerar desconfiança e manipulação. Ganharam bastante foco por distorcerem fatos e negarem a realidade. Como mostra BBC (2019), os responsáveis

por criarem tais notícias não acreditam nelas, mas sabem que existem aqueles acreditam e irão propagá-las.

Praticamente todas as *fake news* não possuem conteúdo verificável, quase sempre possuem títulos sensacionalistas visando atingir o maior número de pessoas pelo compartilhamento. São desenvolvidas de modo a se aproveitar das emoções que são capazes de desencadear nos leitores. Segundo Doretto (2019), alguns jovens não se importam com fontes, onde postagens em redes sociais ou aplicativos são o suficiente quando se trata de garantir a autenticidade da fonte. Neste ambiente é onde surgem algumas notícias que visam atingir grupos fechados, onde a maioria de seus integrantes por algum motivo, seja a falta de vontade em se integrar sobre o assunto ou até mesmo por não saberem/não desejarem atestar a veracidade da notícia compartilhada, acabam por acreditar nelas. O efeito é ainda potencializado através dos filtros personalizados que, com seus algoritmos de relevância, acabam por entender que o usuário deseja acessar cada vez mais este tipo de conteúdo e, portanto, limitam imensamente a percepção do usuário, que entra cada vez menos em contato com ideias e informações afrontam as suas, o induzindo a acreditar que o mundo partilha das mesmas concepções que ele.

2.5 VIÉS DE CONFIRMAÇÃO

Segundo Faber (2014), viés de confirmação é "tendência de concordarmos com pessoas e ideias que concordam com as nossas", ou seja, é a tendência natural de procurar fatos e evidências que reforcem ideias já pré concebidas.

Opiniões e crenças, são resultado de anos de informações que acumuladas reforçaram aquilo que já se encontrava estabelecido, buscando evitar os fatos e provas que poderiam contradizer aquilo que já se tinha estabelecido. Humanos não querem estar errados, por este motivo buscam provas de que estão certos, evitando de maneira consciente ou não, provas contraditórias.

O cérebro é um órgão muito fascinante, de modo imperceptível ele se encontra constantemente fazendo uma série de cálculos, verificando a todo segundo se uma decisão tomada é a melhor em determinado momento. Segundo Pariser (2012), o modo que ele encontra para fazer isso é a todo instante fazer comparações que dizem se determinada ação tomada é compensatória ou não, e ainda mantem um delicado equilíbrio em se aprender demais com eventos passados e ao mesmo tempo aprender com estímulos do presente. O modo como operam os filtros personalizados afetam diretamente nesse equilíbrio, entre fortalecer ideias já existentes e a obtenção de novas. Inicialmente eles cercam o usuário com ideias com as quais ele já encontra familiarizado, o que culmina por gerar um excesso de confiança. Isso acaba por impedir o usuário de entrar em contato com ideias diferentes, esse é um fator determinante que possibilita um

crescimento individual, visto que é no momento em que se entra em contato com outras formas de se ver uma mesma ideia que ocorre um novo aprendizado. Dificilmente o usuário buscará alguma informação que o contradiga, este fato aliado aos filtros personalizados, cria um sistema de retroalimentação onde o usuário acaba por sempre interagir com os mesmos conteúdos com pequenas modificações. Este ambiente que acaba por ser criado é altamente propício a *fake news*, já que são feitas de modo a explorar como funcionam os algoritmos de recomendação (PARISER, 2012).

2.6 AGÊNCIAS DE CHECAGEM DE FATOS

A imensa quantidade de informação disponível hoje, fez com que surgisse pressões no sentido da velocidade em que se publica alguma notícia, onde ser o primeiro a publicar se tornou mais relevante do que checar a veracidade das notícias (GEHLEN, 2011).

Fact-checking é uma técnica de checagem de fatos, que segundo Gehlen (2011), "passou a designar também a atividade de profissionais e/ou de agências dedicados a confirmar informações ou a detectar imprecisões, erros e mentiras nos discursos de personalidades públicas veiculados na mídia".

Segundo Fatos (2021), "A checagem de fatos é um método jornalístico por meio do qual é possível certificar se a informação apurada foi obtida por meio de fontes confiáveis e, então, avaliar se é verdadeira ou falsa, se é sustentável ou não".

Como menciona Gehlen (2011), a agência de jornalismo e checagem Lupa S/A, foi considerada a primeira a realizar este trabalho no Brasil. Segundo Lupa (2015b), "A Lupa é uma plataforma de combate à desinformação através do *fact-checking* e da educação midiática", surgiu em 2015, se especializou na checagem de fatos, ela acompanha os noticiários sobre política economia, busca corrigir informações imprecisas e divulgar dados corretos (LUPA, 2015b).

Aos Fatos, é um site jornalístico de checagem de fatos. Segundo Fatos (2021a), o site realiza acompanhamentos das declarações de políticos e autoridades de expressão nacional, dos mais variados partidos. Ainda segundo Fatos (2021a), realiza um processo em 7 etapas para verificação de uma notícia, sendo: selecionar uma notícia pela relevância, verificar quem divulgou a informação, procurar fontes de origem confiável, consultar fontes oficiais para confirmar ou refutar a informação, onde caso seja necessário são consultadas fontes alternativas, contextualizar e por fim classificar a notícia.

Fato ou *Fake*, serviço de checagem lançado pelo G1. Segundo G1 (2018), o serviço contara com jornalistas que iram monitorar diariamente notícias que veiculam por meio das redes sociais e aplicativos.

No contexto de checagem de fatos, passam a surgir estudos que tem por finalidade definir as habilidades básicas que um usuário deveria possuir ao interagir com ferramentas digitais. A área que se preocupa com isso é denominada Literacia Digital. Segundo Martin (2008), é a capacidade do indivíduo em usufruir das ferramentas digitais, sendo ele capaz de analisar e identificar corretamente informações disponíveis nos meios digitais para a aquisição de novos conhecimentos, sendo ainda capaz de utilizar das mídias sociais de modo construtivo.

Como mostra Santos, Azevedo e Pedro (2015), muitos autores procuram fazer com que o termo Literacia Digital seja mais abrangente, assim unindo as diferentes literacias que tratam dos meios digitais em uma só.

2.7 PROCESSAMENTO DE LINGUAGEM NATURAL

Linguagem natural é a linguagem utilizada no cotidiano, é a língua que as pessoas utilizam para conversar, escrever e transmitir ideias (BIRD; KLEIN; LOPER, 2009). No entanto esta não é a língua que costumeiramente máquinas utilizam para realizarem suas tarefas. Neste cenário crescem as pesquisas na área de Processamento de Linguagem Natural (PLN), que tem como um de seus objetivos fornecer uma interface humano-máquina mais intuitiva.

A PLN é uma área de pesquisa que estuda como computadores podem ser usados para compreender e utilizar linguagem natural contida em textos ou na fala, que permitem a execução de tarefas úteis ao ser humano (CHOWDHURY, 2003). O ramo de pesquisa que estuda a compreensão da linguagem humana, exige conhecimento em diversas áreas, ciência da computação, linguística, entre outras, por este motivo são utilizadas técnicas e ferramentas que já foram desenvolvidas e testadas por pesquisadores destas áreas (ALLEN, 1995).

Tecnologias que utilizam Processamento de Linguagem Natural, já são comuns, fazem parte do dia a dia, são utilizadas nos celulares com a correção automática, nos motores de busca, na tradução automática, que facilitam a interação humano-máquina (BIRD; KLEIN; LOPER, 2009).

O processamento de linguagem natural pode ser dividido em duas abordagens: Abordagem clássica, e Abordagem Estatística.

Apesar do PLN ser dividido em duas abordagens, não significa necessariamente que uma é inferior ou superior à outra. Por exemplo a abordagem estatística requer grandes quantidades de dados rotuladas corretamente no idioma que se deseja utilizar, sendo mais viável quando se dispõe de um vasto conjunto de dados. A abordagem clássica, por outro lado não está tão presa a um idioma específico, pois pode ser atingida por uma sequência de passos já definidas, sendo mais necessário conhecimento sobre a estrutura do idioma que será utilizado. Algumas abordagens hoje, adotam uma implementação híbrida, que utilizam técnicas e ferramentas de

ambas as abordagens (INDURKHYA; DAMERAU, 2010). A seguir as duas abordagens serão detalhadas.

2.7.1 Abordagem clássica

Comumente o processamento de linguagem natural tende a ser decomposto em etapas, sendo elas: Sintaxe, Semântica e Pragmática, onde inicialmente um texto é divido em sentenças onde os termos que o compõe possam ser analisados em virtude de sua sintaxe, que produz uma estrutura mais amigável à análise de semântica e significado. É, então, seguido por uma análise pragmática que tem por propósito avaliar o objetivo de uma palavra ou sentença dentro de seu contexto. Esta divisão de um texto em sintaxe, semântica, e pragmatismo, por vezes pode ser um pouco ambígua ou confusa, e é um consenso que na realidade não é simples aplicar esta estratificação. No entanto, esta divisão é um bom meio de início, servindo mais como um modelo didático e como um modelo de arquitetura de um sistema de processamento de linguagem natural. Atualmente com o conhecimento disponível hoje, é possível refinar este processo de decomposição, ficando assim: tokenização, análise léxica, análise sintática, análise semântica, análise pragmática. Isso demonstra que sobre as partes mais concretas, que envolvem tokenizacao, análise léxica, e sintática, se tem um conhecimento mais robusto, que permite a criação de ferramentas e técnicas mais genéricas, porém nas últimas etapas isso é mais difícil, tendo em vista o caráter ambíguo que muitas palavras ou frases podem ter a depender do contexto (INDURKHYA; DAMERAU, 2010).

2.7.1.1 Pré-processamento de texto

Ao obter qualquer tipo de texto como entrada, é necessário realizar algum trabalho de limpeza neste texto, para que fique mais simples a sua segmentação para posterior análise.

Segundo Indurkhya e Damerau (2010), para a realização de uma análise eficiente inicialmente é preciso definir claramente quais são os caracteres que compõem o texto, bem como suas palavras e sentenças. É necessário possuir estes componentes nitidamente bem definidos. Nesta etapa pode ser necessário realizar um processo de limpeza, que consiste em identificar a codificação do texto e convertê-la para codificação a ser usada, remover do texto imagens, links, tags HTML, ou qualquer coisa que não agregue valor ao texto em si (INDURKHYA; DAMERAU, 2010).

O Pré-processamento de texto é uma etapa essencial em qualquer sistema de Processamento de Linguagem Natural, já que as palavras e sentenças são a base para as etapas subsequentes (INDURKHYA; DAMERAU, 2010).

2.7.1.1.1 Tokenização

A quebra do texto em sentenças e a quebra de sentenças em palavras, definindo onde uma palavra termina e começa; na linguística computacional, as palavras identificadas assim são denominadas tokens, e este processo de segmentação é chamado tokenização (INDURKHYA; DAMERAU, 2010).

Em idiomas em que a maioria das palavras são delimitadas por inserções de espaços em branco, é mais simples fazer a segmentação do texto em tokens.

Mas mesmo em idiomas onde as palavras são delimitadas por espaços em branco, existem algumas dificuldades no momento da tokenização. Sinais de pontuação como: pontos, vírgulas, aspas, hifens, e outras marcações, geram ambiguidade no momento da tokenização, pois eles podem delegar funções diferentes a uma frase. Por este motivo o tokenizador utilizado deve estar apto a utilizar os sinais de pontuação e ser capaz de determinar quando um sinal faz parte de um token e quando é símbolo (INDURKHYA; DAMERAU, 2010).

2.7.1.1.2 Segmentação de Frase

Na grande maioria das línguas, as frases são delimitadas por sinais de pontuação, porém nem sempre as suas regras de uso são bem delimitadas, o que faz com que o segmentador de frases a ser utilizado possua um entendimento de seus mais diversos usos. A complexidade de execução desta etapa varia bastante a depender do idioma utilizado (INDURKHYA; DAMERAU, 2010).

Com isso a definição do que constitui uma sentença tende a ser arbitrária, ficando muitas vezes a cargo do desenvolvedor do sistema definir quais são as regras a serem utilizadas para a definição dos limites de uma frase. No entanto a grande maioria dos sistemas de processamento de linguagem natural utiliza como delimitação dos limites de uma frase, um método que consiste em verificar espaços seguidos por uma palavra iniciada em letra maiúscula seguindo até o ponto final, ou ponto de exclamação, interrogação, entre outros (INDURKHYA; DAMERAU, 2010).

2.7.1.2 Análise Léxica

Análise léxica está relacionada à morfologia das palavras, já que é necessário possuir um entendimento sobre a estrutura que forma as palavras.

A análise léxica pode conter diversas etapas, que podem conter regras de extração que prioriza a ocorrência de palavras entre aspas dentro parênteses, entre outras.

Os processos mais comuns que são feitos nesta etapa são: *lemming*, e *stemming*.

- *lemming*: é o processo de relacionar as mais diferentes ocorrências morfológicas de uma palavra em uma única forma, que seria a forma mais básica de uma palavra contida no dicionário, ou seja, seu radical, isso é denominado *lema*. Exemplo: tiver, tenho, tinha, tem são do mesmo *lema* ter (INDURKHYA; DAMERAU, 2010).

- *stemming*: é um processo mais bruto, ele remove o final de uma palavra com o objetivo de reduzí-la a uma forma mais básica, para que seja possível encontrar a ocorrência dessa mesma palavra em diferentes formatos ao longo do texto (STANFORD, 2008).

Stemming e *lemming* compartilham o mesmo objetivo, que é reduzir uma palavra à sua forma mais básica. No entanto a utilização de um processo de lematização, requer mais processamento e geralmente necessita de ferramentas adicionais que lidem unicamente com isso, fazendo com que o processo de *lemming* seja mais viável em sistemas mais robustos. Já para aplicações mais simples é preferível a utilização do *stemming*, ele é mais rápido, porém sua utilização pode ocasionar perda em palavras dependo do que for removido, já que é um método mais bruto.

No desenvolvimento do sistema, foi dada preferência a utilizado do processo de *stemming*, tendo em vista que é um processo mais simples e requer menos processamento, mais viável em sistemas mais simples.

2.7.1.3 Análise Sintática

Esta etapa realiza uma análise gramatical, essa análise é feita sobre uma sequência de palavras fornecidas, comumente sendo uma frase, ela gera uma estrutura de acordo com a gramática utilizada, e esta estrutura é utilizada posteriormente a fim de atribuir algum significado a frase ou palavra. Nesta etapa as palavras fornecidas habitualmente terão sido as palavras processadas na etapa de tokenização, e na etapa de análise léxica. Nesse momento também é feita a atribuição de tags, tag é a atribuição da palavra a alguma classe gramatical, as tags são úteis para a extração de alguma informação pertinente do texto (INDURKHYA; DAMERAU, 2010).

2.7.1.4 Análise Semântica

Nesta etapa é realizado o processamento em cima da sentença já tratada pelas etapas anteriores, com o propósito de "compreender"o seu significado. Dependendo da implementação desta etapa aqui também é feita a extração de informação, que busca adquirir algum conteúdo relevante para usuário, sem que ele tenha a necessidade ler ou compreender todo o conteúdo

que um texto pode fornecer. Esta etapa também pode ser considerada uma análise pragmática, já aqui procura-se também compreender o significado da sentença fornecida.

É nesta etapa que também dependendo da implementação do sistema em questão, pode ser modelada para a realização de resumos automáticos, mineração de dados e tradução automática.

Ela ainda possui utilização para auxiliar na ontologia da internet. Segundo Gruber (1993), ontologia é a especificação de uma conceituação, é a definição formal de forma explícita de propriedades e restrições. Com esta definição, ontologia seria um modelo de dados que se deseja representar. Neste contexto surge a Web Semântica que tem por objetivo fazer com que as páginas da rede possuam uma semântica clara e definida, que visa facilitar a obtenção de informação sem o uso de técnicas avançadas de busca (FREITAS; SCHULZ, 2009).

Conforme a análise semântica passa a se tornar mais robusta e refinada, isto torna mais simples a compreensão dos dados de entrada fornecidos pelo usuário, que torna mais eficiente a interação humano-máquina (INDURKHYA; DAMERAU, 2010).

2.7.2 Abordagem Estatística

No campo do processamento da linguagem natural, a abordagem estatística se tornou mais preponderante. A abordagem estatística para o processamento de linguagem natural utiliza muito de técnicas de aprendizado de máquina, onde para desenvolver um sistema capaz de lidar com linguagem natural são utilizadas de vastos conjuntos de dados, que são fornecidos a um algoritmo que procura padrões dentro do conjunto fornecido, os padrões encontrados então passam a fazer parte de um modelo que possui a capacidade de compreender dados de entrada fornecidos em linguagem natural (INDURKHYA; DAMERAU, 2010).

A abordagem estatística se tornou mais popular por não necessitar de conhecimentos tão especializados, pois não exige uma análise tão profunda, bastando muitas vezes possuir um conjunto de dados bastante robusto e uma correta classificação dos dados utilizados no treinamento.

2.8 CONSIDERAÇÕES

Nas seções anteriores foram apresentados os conceitos que fundamentam o sistema. Sendo o PLN a base na qual o sistema foi desenvolvido, pois consiste em analisar a notícia fornecida pelo usuário, e lhe fornece notícias similares, e para a realização deste processamento são utilizadas ferramentas que lidam com a linguagem natural. Também foram apresentados os motivos que levaram ao desenvolvimento deste trabalho, como é possível observar nas seções iniciais do capítulo 2, o excesso de informação e perda de credibilidade dos meios de notícias tradicionais,

levaram à onda de negacionismo munida de *fake news*. Na tentativa de minimizar os estragos ocasionados pelas *fake news*, foi desenvolvido o sistema, que mostra ao usuário notícias similares com aquele fornecida por ele. A solução fornecida por este trabalho é paliativa, pois o sistema trata os efeitos de algo que pode ser observado, mas a causa continua intocada.

2.9 FERRAMENTAS UTILIZADAS

A elaboração de um sistema que utiliza PLN, poder ser feita de diversas formas, e em ampla gama de linguagens de programação. Este trabalho optou por utilizar ferramentas feitas predominantemente com Python. O Python foi escolhido por ser uma linguagem relativamente simples de mexer, e já possuir um amplo ecossistema de bibliotecas que lidam com processamento de dados, o que facilitou o desenvolvimento do trabalho.

2.9.1 Python

Python é uma linguagem de programação de alto nível, interpretada e interativa orientada a objetos. Segundo Pedregosa et al. (2011), Python está se estabelecendo como uma das línguas de programação mais populares para a computação científica. Ainda como afirma Pedregosa et al. (2011), por sua natureza interativa e seu ecossistema de bibliotecas que lidam com processamento de dados, tem feito com que esta linguagem se torne atraente para quem deseja trabalhar com a análise de dados. Por este motivo foi a linguagem de programação escolhida para o desenvolvimento deste trabalho.

O Python disponibiliza algumas estruturas de dados que facilitam no momento da codificação, uma estrutura que vale uma breve menção é o dicionário. Um exemplo de dicionário em Python, pode ser visualizado na Figura 1.

Figura 1 – Exemplo de um dicionário em Python

```
#exemplo de dicionário que armazena os dados de uma pessoa
pessoa = {
    "nome" : "Carcaju",
    "idade": 50,
    "cpf"  : 000000000
}
#exemplo de como se recupera um dado contido em dicionário
nome = pessoa['nome'] #aqui é recuperado o dado 'nome', no caso, 'Carcaju'
```

Fonte: Elaborado pelo autor, 2021.

Dicionários, cujo tipo é *dict*, é uma estrutura semelhante a um "vetor associativo", onde a indexação dos dados é feita por meio de chaves, que são associadas a um valor. As chaves

podem ser qualquer tipo imutável, *strings* e inteiros que designam algum valor (PYTHON, 2021).

Dicionários são delimitados por chaves, e contém uma lista de pares chave/valor delimitada por uma vírgula. As principais operações realizadas em um dicionário são: armazenar, e recuperar valores a partir das chaves (PYTHON, 2021). Figura 1.

2.9.2 Telegram

O Telegram é um aplicativo de comunicação semelhante ao WhatsApp, que se tornou bastante popular atualmente. O Telegram disponibiliza uma API para criação de bot, estes bots podem ser usados para diversas atividades, dentre elas, o gerenciamento de grupos e conversas, o que permite a criação de uma solução automatizada para os mais diversos problemas (TELEGRAM, 2021).

Para a interação com esta api disponibilizada pelo Telegram, foi utilizada a biblioteca python-telegram-bot, que fornece uma interface em Python puro, para a manipulação da API do Telegram (PYTHONTELEGRAMBOT, 2021).

2.9.3 Heroku

Heroku é uma plataforma em nuvem para *deploy* de aplicações. O Heroku fornece um conta gratuita com a qual é possível disponibilizar uma aplicação, muito útil para testar sites e executar aplicativos na fase de teste. Por este motivo ele foi escolhido como servidor para *deploy* da aplicação (HEROKU, 2021). Como foi utilizada uma conta gratuita, a aplicação não fica ativa a todo momento, após um período de inatividade a aplicação entra em um estado de dormência, caso a aplicação seja requisitada enquanto neste está de dormência ela tem uma demora em fornecer uma resposta, que pode levar 10min, uma vez que ela entre em estado de atividade ela não terá mais essa demora na resposta.

2.9.4 Ferramentas de Processamento de Linguagem Natural

Existem várias ferramentas para se utilizar em sistema que vise executar alguma tarefa que envolva o Processamento de Linguagem Natural. Neste trabalho foram testadas duas ferramentas específicas, sendo elas: NLTK, e SpaCy. Estas duas ferramentas foram escolhidas por três motivos, primeiro elas já possuem algum tipo de tratamento para o português, segundo são relativamente simples de se mexer, terceiro são em Python. Também foi testado o NLPyPort,

ferramenta desenvolvida em cima do NLTK para o processamento de linguagem natural em português.

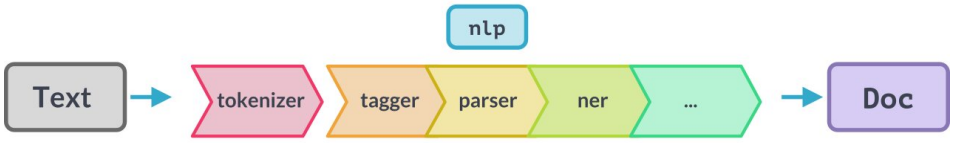
2.9.4.1 *Tagger*

O *tagger* é um categorizador, ele inclui cada palavra a sua respectiva classe gramatical. Utilizado depois do pré-processamento, afim de extrair alguma informação que possa a vir ser relevante, como por exemplo, nomes próprios contidos em um texto. Ele não necessáriamente é uma ferramenta de PLN, mas um componente presente em uma de suas etapas, sendo algo comum a todas as ferramentas aqui descritas. Com exceção do Spacy e NLPyPort, para as outras ferramentas aqui descritas é fornecido um *tagger* treinado em um corpus em português.

2.9.4.2 SpaCy

É uma biblioteca de código aberto utilizada para PNL. Spacy foi projetado para ser utilizado em produção para o processamento de grandes volumes de texto. Pode ser usado para a extração de informação ou de compreensão de linguagem natural. .

Figura 2 – *Pipeline* de execução do SpaCy



Fonte: (SPACY, 2021).

O SpaCy funciona com *pipelines* de processamento, que são etapas de processamento feitas em cima do texto fornecido. Ao final de uma *pipeline*, é devolvido um objeto doc que contém os dados provenientes do processamento do texto. Figura 2.

SpaCy é simples de se utilizar, e já possui uma *pipeline* em português que pode ser utilizada.

2.9.4.3 *Natural Language Toolkit* (NLTK)

Esta ferramenta para o processamento de linguagem natural é mais complexa, exigindo um conhecimento maior por parte do usuário. Ao contrário do SpaCy, o NLTK não possui uma *pipeline* já pronta, sendo necessário ao usuário chamar os métodos necessários para a utilização efetiva, bem como já ter os dados tratados para cada etapa a ser utilizada.

2.9.4.4 NLPyPort

É uma ferramenta desenvolvida para a análise de linguagem natural desenvolvida para o português (FERREIRA, 2019). Esta ferramenta apresentou algumas desvantagens, como: carência de documentação sobre sua utilização, alguns problemas de incompatibilidade, que afetavam o *tagger* e o NLTK, e o fato de executar uma *pipeline* única, dificultando sua utilização modular. Por estes motivos essa ferramenta não foi utilizada. Para a utilização das etapas de processamento, foi optado pela utilização do NLTK puro. Pois com NLTK é possível segmentar a etapa de precessamento como se deseja.

2.9.4.5 Considerações

O SpaCy apesar de ser simples de ser utilizado, requer que toda a vez que for chamado execute todo o *pipeline*, o que nem sempre é necessário para um sistema que não vá trabalhar com grandes volumes de dados acabe por não ser tão vantajoso. Isso aliado ao fato de não conseguir chamar métodos independentes, fez com que a escolha dessa ferramenta fosse descartada.

O NLPyPort apesar de ser uma proposta interessante, que já foi tratada para o idioma português, não foi utilizado por apresentar alguns pontos negativos que já foram mencionados antes.

Por fim, o NLTK, foi escolhido por permitir sua utilização de forma modularizada, onde pode-se chamar partes especificas do código sem ter que executar todo um processamento antes.

2.9.5 Aprendizado de máquina

Segundo Müller e Guido (2016) Aprendizado de máquina é um método de análise de dados, com o objetivo de extrair algum conhecimento que possa vir a estar contido ali. É um campo de pesquisa que une diferentes áreas do conhecimento, estatística, inteligência artificial, e ciência da computação. Algoritmos de aprendizado de máquina possuem a incrível capacidade de encontrar padrões. O que viabilizou a utilização de algoritmos de aprendizado de máquina foi a imensa quantidade de dados que é produzida, que como mostra Rydning (2018), em 2025 pode chegar a 175 ZB, neste contexto surge o *Big Data*. Segundo Rautenberg e Carmo (2019), *Big Data*, pode ser entendido como "um conceito que caracteriza volumosos conjuntos de dados heterogêneos, os quais não são passíveis de processamento por soluções computacio-

nais tradicionais", o que caracteriza um campo fértil para o aprendizado de máquina, que pode vir a tirar informações ou conhecimentos destes conjuntos de dados.

Apesar do aprendizado de máquina possuir diversas aplicações, hoje é massivamente utilizado para fazer com que seja dedicado cada vez mais tempo em redes sociais, e a elaboração de *marketing* altamente direcionado, com o intuito de fazer o usuário comprar produtos definidos de acordo com suas preferência pessoais.

2.9.5.1 TF-IDF

Segundo Aizawa (2003), Termo de Frequência - Inverso da Frequência nos Documentos (TF-IDF), é um método de ponderação muito utilizado em PLN para a recuperação de informação. Como mostra Ramos et al. (2003), ele determina a frequência relativa de palavras em um documento específico, em comparação com a proporção inversa desta palavra em todos os documento fornecidos.

2.9.5.2 Sklearn

Sklearn é uma biblioteca em Python que integra uma gama de algoritmos de aprendizado de máquina , que oferece um uso mais facilitado, possui um bom desempenho, e possui uma documentação bem abrangente (PEDREGOSA et al., 2011).

Neste trabalho o Sklearn foi utilizado para verificar a similaridade entre textos, e em uma implementação simples de um sumarizador automático. Ele foi utilizado para verificar a similaridade entre textos pois já possui uma série de métodos que juntos podem ser utilizados para a verificação de similaridade. Foi utilizado no auxílio da implementação de um sumarizador simples, pois já possui uma implementação do TF-IDF, sendo necessário apenas adaptar seu código a ele.

2.9.5.3 FastText

É uma biblioteca de código aberto, desenvolvida pelo *Facebook*, com ela é possível o desenvolvimento de modelos de classificação de texto, que é treinado a partir textos rotulados. Segundo Facebook (2020), permite a criação de modelos por treino supervisionado e treino não supervisionado, eu utilizei apenas o treino supervisionado, onde foi fornecido dados rotulados e fornecidos ao modelo para treino.

2.9.6 *Web Scraping*

Web Scraping é a prática de interagir com a internet de modo mais programático, onde ao invés de ser um humano interagindo com a web, é um programa. *Web Scraping* tem por objetivo a coleta de dados, e sua posterior persistência em um banco de dados. Este processo segue alguns passos, que envolve ir a uma página, obter todo o conteúdo desta página. Este conteúdo então passa por um processo de limpeza, onde é removido o que não se deseja, após esta limpeza os dados são persistidos. Basicamente *Web Scraping* consiste em coletar, limpar, e armazenar dados, a complexidade de cada etapa varia de acordo com a dificuldade da obtenção de dados (MITCHELL, 2018).

2.9.7 *Web Crawling*

Pode ser definido como um bot da internet que navega sistematicamente pela rede. Um exemplo de utilização bem famoso deste tipo de bot é o Google, ele utiliza destes bots para fazer a indexação de páginas em seu mecanismo de busca. Este tipo de bot é chamado comumente de *Spider*.

2.9.7.1 Selenium

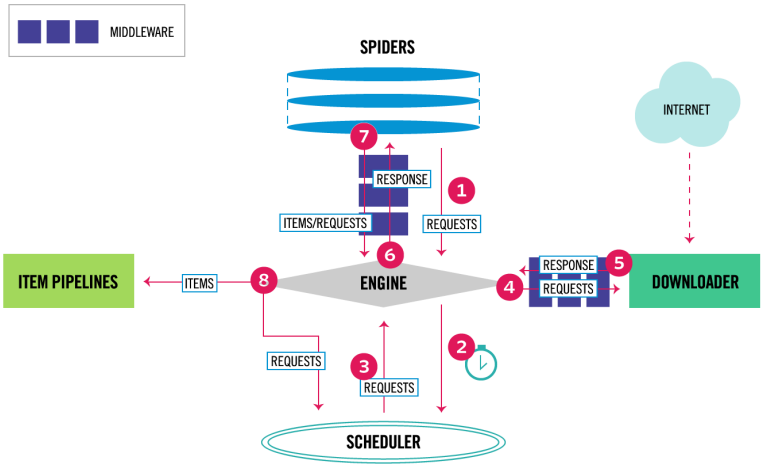
Selenium é uma ferramenta utilizada para realização de testes automatizados em navegadores web, ele disponibiliza ferramentas que possibilitam emular a interação do usuário em um navegador (SELENIUM, 2021).

O fato de o Selenium possibilitar a emulação da interação do usuário com web, o torna uma ferramenta que pode ser utilizada para mais funções além da execução de testes automatizados. Com o Selenium, também é possível a realização *web scraping*. No entanto a utilização do Selenium para *web scraping*, é mais complexa, pois como o Selenium não foi concebido como uma ferramenta de *scraping*, é necessário desenvolver toda uma lógica que utilize ele para executar a tarefa desejada.

2.9.7.2 Scrapy

Scrapy é um *framework* de código aberto para a coleta de dados na web a partir de sites. Na Figura 3, é possível observar a arquitetura do Scrapy.

Figura 3 – Arquitetura Scrapy



Fonte: (SCRAPY, 2020).

O Scrapy é composto pelos seguintes componentes:

- *Engine*: Controla o fluxo de dados entre os componentes, dispara eventos dependendo das ações ocorridas.
- *Scheduler*: Escalonador, gerencia a ordem que devem ocorrer os processos, os ordenando em filas.
- *Downloader*: Busca as páginas para posteriormente fornecê-las as *spiders*.
- *Item Pipeline*: Processos a serem executados após os *spiders* terminarem de coletar seus conteúdos.
- *Middlewares*: Mecanismo utilizado para fazer algum tratamento dos dados entre o *Downloader* e a *Engine*, e entre as *Spiders* e a *Engine*

A implementação de um sistema próprio de *scraping*, pode ser interessante caso se deseje ter uma melhor compreensão de como este processo decorre até que forneça um resultado visível na tela. No entanto ao optar por desenvolver uma solução própria, acaba-se por descobrir que muitos sites possuem sistemas que dificultam a operação de bots em suas páginas. Na solução próprio seria necessário desenvolver um método genérico que possibilitasse a execução do bot, ou seja, é necessário compreender todas as formas de medidas contra bot existentes e desenvolver contramedidas. O Scrapy por ser um *framework*, já vem com métodos para lidar com estes obstáculos, por isso foi optada sua utilização.

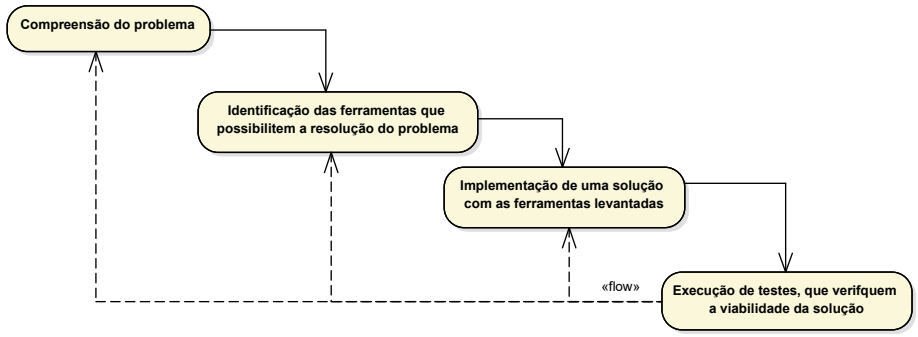
3 DESENVOLVIMENTO

Neste trabalho foi desenvolvido um sistema que fornece uma lista de notícias similares, baseada na notícia fornecida previamente pelo usuário. A seguir, são detalhadas as fases do desenvolvimento desse sistema.

3.1 PROCESSO DE DESENVOLVIMENTO

O projeto foi desenvolvido utilizando de um modelo iterativo incremental, executado em fases. Inicialmente, era realizada a compreensão do problema, procurando entender o que seria necessário resolver. Posteriormente, eram identificadas as ferramentas necessárias na resolução do problema. Subsequentemente, era realizada a implementação da solução, utilizando as ferramentas levantadas. Finalmente, a solução era testada, para verificação de sua viabilidade. Este processo foi executado algumas vezes. A Figura 4 apresenta detalhadamente o processo.

Figura 4 – Modelo iterativo incremental do projeto



Fonte: Elaborado pelo autor, 2021.

Inicialmente, foi feita uma análise buscando compreender o que o sistema deveria fazer, foi identificado que o sistema deveria fornecer uma interface acessível ao usuário e realizar um processamento sobre os dados de entrada fornecidos pelo usuário. Neste ponto foi identificado que ferramentas que lidassem com PLN, seriam utilizadas, e que também seria necessária alguma ferramenta que possibilitasse a interação com o usuário.

Para a realização dos pontos levantados, foi realizada uma busca por ferramentas que possibilitassem a sua implementação.

Por fim, para cada uma das possíveis ferramentas a serem utilizadas na elaboração deste sistema, foi realizada uma POC, avaliando sua viabilidade de uso.

As ferramentas escolhidas para possibilitar a interação com usuário foram: TKinter, PyGame e Telegram. Tkinter e PyGame, fornecem os meios necessários para o desenvolvimento

de suas próprias telas, porém o desenvolvimento dos componentes na tela se mostrou mais complexo do que foi cogitado inicialmente, e para isso exigiriam mais tempo de estudo e teste, por este motivo foram descartadas. Telegram, apesar de não fornecer a possibilidade de criar suas próprias telas e os componentes que à compõem, fornece uma API muito robusta e bem documentada, que possibilita a criação de um bot com a capacidade de interagir com o usuário, além de possuir milhões de usuários ativos o que aumenta o possível alcance da ferramenta. Desta forma, foram escolhidas para possibilitar a interação com a internet: Selenium, Requests e Scrapy. O Selenium necessita de um componente externo ao Python e não fornece nenhum meio facilitado de acessar a rede, sendo necessário desenvolver uma solução própria para tirar proveito dele, por este motivo não foi escolhido. Scrapy, apesar de possuir uma complexidade de utilização maior, foi escolhido por ser um *framework* dedicado à realização de *scraping*, o que facilitava bastante no desenvolvimento. Quanto ao Requests, ele não foi utilizado no projeto final pois o Scrapy cobria suas funcionalidades.

3.1.1 *Backlog*

Para o gerenciamento das atividades foram levantados itens de *backlog*, a serem realizados durante o andamento do projeto. Posteriormente, cada item foi decomposto em atividades menores, que foram executadas em sequência. A seguir são apresentadas as tarefas que formam o *backlog*.

1. Definição de uma tecnologia a ser utilizada na classificação de texto.
    (a) Buscar ferramentas que viabilisem, a utilização de dados, para a contrução de um modelo de classificação.
    (b) FastText
        i. Estudar conceitos básicos
        ii. Desenvolvimento de uma POC
    (c) Encontrar conjunto de dados sobre *fake news*, já rotulados em português do Brasil.
    (d) Desenvolver um algoritmo de pré-processamento dos dados antes de serem fornecidos ao Fasttext, baseado na validação cruzada *Holdout*.
    (e) Verificar resultados fornecidos pelo modelo.
    (f) Modificar algoritmo de pré-processamento, acrescentando sumarização de texto.
    (g) Avaliar resultado do modelo.
2. Definir a tecnologia a ser utilizada no módulo de PLN (Spacy, NLTK).

(a) Spacy

i. Estudar conceitos básicos.

ii. Analisar exemplos de utilização.

iii. Desenvolvimento de uma POC, aplicando os recursos da ferramenta.

(b) NLTK.

i. Estudar conceitos básicos.

ii. Analisar exemplos de utilização.

iii. Desenvolvimento de uma POC, aplicando os recursos da ferramenta.

(c) Escolher a ferramenta a ser utilizada.

3. Definir se a coleta de dados será feita utilizando um *framework*, ou se será adotada uma solução própria.

(a) Pesquisar tecnologias que podem ser utilizadas.

(b) Desenvolver uma POC com Selenium.

i. Acessar página principal do Google.

ii. Enviar parâmetros de busca para o Google.

iii. Coletar conteúdo da página de retorno inicial da busca.

iv. Obter todos os links da página de resposta inicial do Google.

v. Acessar todos os links obtidos da página inicial.

vi. Obter todo o conteúdo das tags "p"dos sites acessados.

vii. Fazer a execução do navegador ser executa em segundo plano.

viii. Avaliar resultado do projeto.

(c) Desenvolver uma POC utilizando requests.

i. Definir lista inicial de urls a serem acessadas.

ii. Fazer requisição do conteúdo das urls iniciais.

iii. Utilizar o pacote Python *Beautiful Soup*, para parsear o conteúdo obtido das páginas acessadas.

iv. Coletar os textos contidos nas tag "p"das páginas.

v. Definir um tempo de expiração para a requisição (*timeout*) das páginas. Utilizar também tratamento de erros.

vi. Avaliar resultado do projeto.

(d) Desenvolver uma POC utilizando Scrapy.

i. Definir lista inicial de urls a serem acessadas.
ii. Compreender o funcionamento básico do Scrapy utilizando o tutorial fornecido pelo próprio Scrapy.
iii. Desenvolver um *spider* que acessa o G1.
iv. Executar o *spider*, verificar se esta trazendo o conteúdo do site correspondente.
v. Analisar a estrutura do site do G1, compreender como estão estruturadas as notícias.
vi. Parsear o conteúdo do site, obter todos os links que levam para a notícia.
vii. Seguir todos o links recuperados.
viii. Obter o conteúdo das tags "p", contido no elemento com a propriedade "itemprop=articleBody".
ix. Analisar documentação do Scrapy, verificar como executar *spiders* pelo *script*, e como passar parametros para os *spiders*.
x. Executar *spider* pelo código.
xi. Definir os *pipelines* a serem executados após os *spiders* finalizarem sua busca nos sites.
xii. Avaliar resultado do projeto.

4. Desenvolver uma aplicação de teste utilizando Telegram.
   (a) Estudar a api do Telegram em python.
   (b) Colocar a aplicação de teste no ar.
   (c) Testar bot no Telegram.
5. Desenvolvimento de um protótipo unindo todas as tecnologias utilizadas.
   (a) Criar um novo projeto em Python.
   (b) Juntar o projeto do Telegram com o projeto do Scrapy.
   (c) Corrigir problema de não poder reiniciar o twisted, ocasionado pois ele deve ser executado na *thread* principal, e só deve ser inicializado uma vez no projeto.
   (d) Ajustar a estrutura de pacotes no projeto.
   (e) Na mensagem inicial mostrada, incluir o nome do usuário.

3.2 ESPECIFICAÇÃO DOS REQUISITOS

Nesta seção serão descritos os requisitos a serem implementados no analisador de *fake news*.

3.2.1 Requisitos Funcionais

RF1: O sistema deve permitir a inclusão de texto (**RN1**).

RF2: O sistema deve realizar processamento sobre o texto fornecido. Quebrar texto em sentença, sentença em palavras (**RN2**).

RF3: O sistema deve extrair a frase mais importante do texto informado, desta frase adquirir o conteúdo para realizar uma procura (**RN2, RN3**).

RF4: O sistema deve procurar notícias semelhantes em sites pré-definidos (**RN2, RN3**).

RF5: O sistema deve fornecer um *feedback* para que o usuário saiba que o sistema está realizando alguma atividade (**RN2, RN3**).

RF5: O sistema deve fornecer um rótulo a notícia, utilizando um modelo de classificação já treinado (**RN4**).

RF6: O sistema deve mostrar ao usuário as notícias semelhantes encontradas, bem como o link para acessá-las (**RN2, RN3**).

3.2.2 Requisitos não funcionais

RNF1: O sistema deverá fornecer uma interação com o usuário através do Telegram.

RNF2: O sistema deverá utilizar o *framework* Scrapy para realizar a procura por notícias.

RNF3: O sistema deverá utilizar o conjunto de ferramentas contido no NLTK para a realização do processamento em cima do texto fornecido.

RNF4: O sistema deverá utilizar o Sklearn para realizar a verificação de semelhança entre strings.

RNF5: Para o desenvolvimento de um modelo capaz de classificar um texto fornecido como *fake* ou não, deverá ser utilizada a biblioteca desenvolvida em Python Fasttext.

3.2.3 Regras de negócio

RN1: O texto informado deverá possuir mais de 140 caracteres e não deve exceder 500 (**RF1**).

RN2: A quantidade máxima de notícias retornado ao usuário ao fim da execução do programa deve ser 5 (**RF2, RF3, RF4, RF6**).

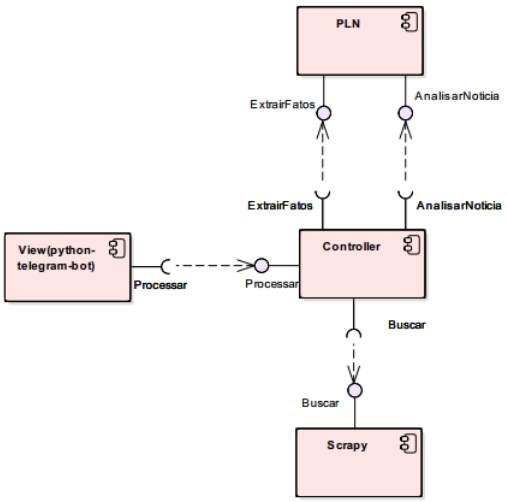
RN3: Caso o sistema não encontre notícias similares isso deve ser informado ao usuário, bem como locais onde ele pode tirar suas dúvidas (**RF3, RF4, RF5, RF6**).

RN4: Deve ser retornada para a notícia fornecida pelo usuário um rótulo, bem como o percentual de confiança que o modelo possui sobre a classificação da notícia.

3.3 PROJETO DO SISTEMA

O sistema foi desenvolvido utilizando uma arquitetura de componentes, Controller, PLN, Scrapy, View.

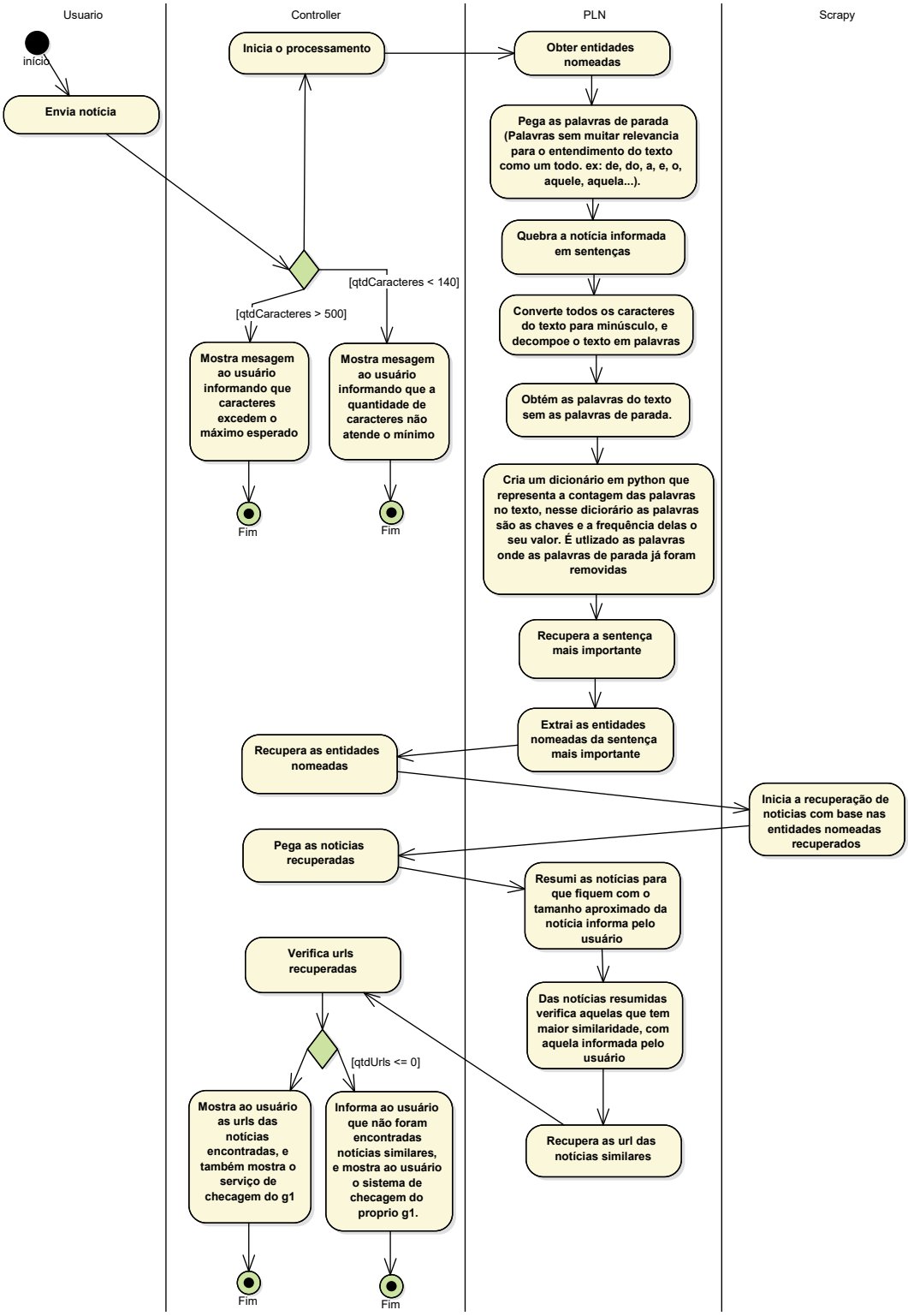
Figura 5 – Diagrama de componentes do projeto



Fonte: Elaborado pelo autor, 2021.

Na Figura 5, é possível observar os diferentes componentes que fazem parte do sistema e as principais ligações entre eles.

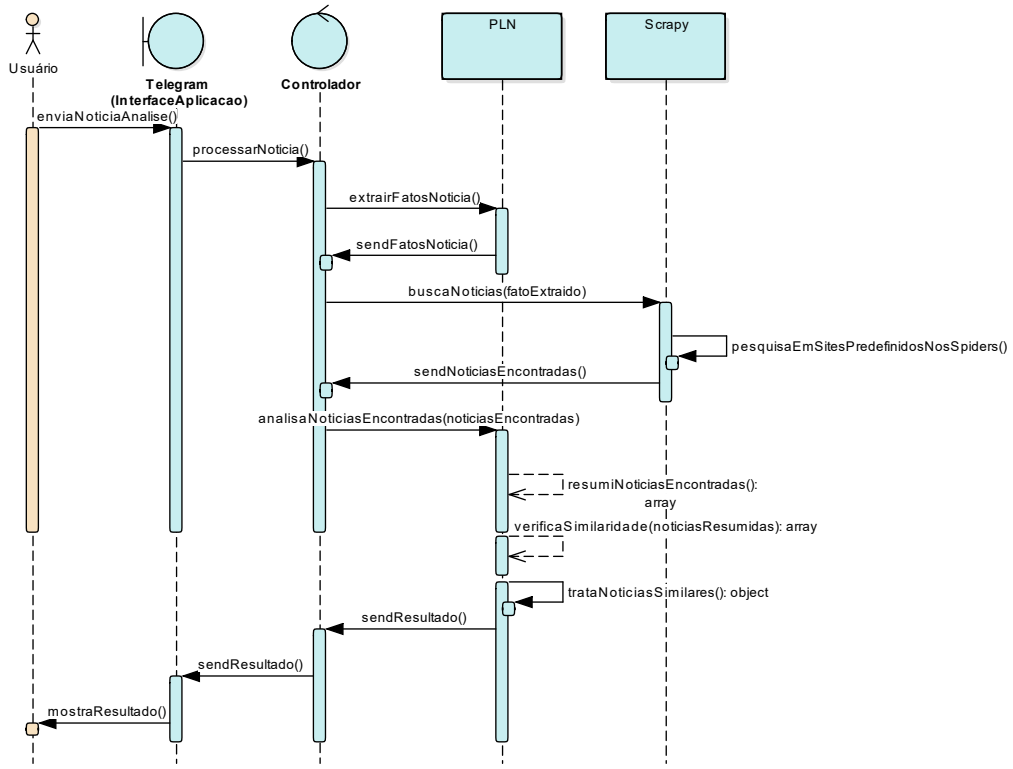
Figura 6 – Fluxo de execução das atividades do sistema



Fonte: Elaborado pelo autor, 2021.

Na Figura 6, é demonstrado os processos que ocorrem no decorrer de uma requisição recebida pelo sistema. O termo entidades nomeadas, que aparece na Figura 6, segundo Amaral (2013), se refere a "termos que apresentam um ou mais designadores rígidos, num determinado texto", os mais comuns são, substantivos próprios, organizações e entidades locais.

Figura 7 – Fluxo do Sistema



Fonte: Elaborado pelo autor, 2021.

Na Figura 7, é detalhada o fluxo de exeção do sistema e os dados que são trocados entre seus diferentes componentes.

3.3.1 Módulo Controller

O Controller fica responsável por gerenciar a comunicação entre os diferentes módulos, repassando e recebendo informações.

3.3.2 Módulo PLN

O módulo de PLN fica responsável por tratar todo o processamento que é feito em cima do texto repassado pelo usuário. Ao final de todo o processamento é retornado um *dict*, contendo informações pertinentes à análise.

3.3.3 Módulo Scrapy

Como já mencionado anteriormente, o Scrapy, é um *framework* desenvolvido em Python para facilitar o processo de *scraping* na internet.

3.3.4 View

No módulo de view foi utilizado o Telegram, por ter muitos usuários, e fornece muitas possibilidades de desenvolvimento.

3.4 IMPLEMENTAÇÃO DO SISTEMA

3.4.1 Estrutura do Controller

O controller fica responsável por fazer as validações iniciais. Nele é carregado tudo aquilo que necessário para o funcionamento da aplicação.

Inicialmente é mostrada uma mensagem de boas-vindas ao usuário. Está é uma mensagem bem simples contendo o nome do usuário que está interagindo com o sistema e uma breve descrição da tarefa que o sistema irá desempenhar. O nome do usuário que aparece é o nome público, pelo qual ele pode ser localizado utilizando a busca do Telegram.

Após o sistema enviar a mensagem de boas-vindas para o usuário, ele inicia a primeira etapa de análise, que consiste em obter as palavras mais relevantes da frase mais importante obtida do texto. Com as palavras obtidas o módulo do Scrapy é iniciado, ele irá obter as notícias parecidas com base nas palavras recuperadas. Com as notícias recuperadas, mais uma vez é utilizado o módulo PLN, que obtém as 5 notícias mais similares. Nesta etapa é onde também é rotulada a notícia fornecida pelo usuário como falsa ou verdadeiro, e mostra a porcentagem de confiança no modelo. Após a coleta e análise destas informações, resultado é apresentado ao usuário.

3.4.2 Estrutura do módulo PLN

Módulo responsável pelo processamento a ser realizado no texto fornecido pelo usuário. Este módulo é acessado duas vezes ao decorrer do andamento do sistema. Ele só é acessado pelo controller. Inicialmente ele é requisitado para a obtenção das palavras mais importantes a serem usadas no módulo do Scrapy, posteriormente ele é chamado para verificar similaridade

entre as notícias encontradas e a notícia do usuário, bem como fornecer um rótulo para a notícia do usuário.

3.4.2.1 Obtenção das palavras mais importantes

O processo de obtenção das palavras mais importantes pode ser decomposto em algumas etapas, sendo elas: Obtenção do texto do usuário, Obtenção das palavras de parada, Quebra do texto em sentenças, Quebra do texto em palavras, Remoção das palavras de parada, criação de um dicionário de frequência das palavras, e a obtenção da sentença mais importante.

Figura 8 – Código utilizado para obter as palavras mais importantes

```
306         pipeline = Pipeline()
307
308         txt       = texto#texto forneceido pelo usuario
309         stops     = pipeline.get_stop_words()
310         sentencas = pipeline.sent_tokenize(txt)
311         palavras  = pipeline.tokenize(txt.lower())
312         #remove stop words as palavras
313         palavras_sem_stops  = [plvr for plvr in palavras if plvr not in stops]
314         frequencia          = pipeline.get_dict_ocorrencia_palavras(palavras_sem_stops)
315
316         ################################
317         #sentencas importantes
318         ranking = defaultdict(int)
319         #ranqueia as sentenças
320         for i, sentenca in enumerate(sentencas):
321             for palavra in pipeline.tokenize(sentenca.lower()):
322                 if palavra in frequencia:
323                     ranking[i] += frequencia[palavra]
324
325         idx_sentencas_importantes = nlargest(1, ranking, key=ranking.get)
326         #aqui eu pego a sentenca mais importante, de acordo com meu sistema de rankeamento
327         sentenca_mais_importante  = sentencas[idx_sentencas_importantes[0]]
328         #aqui eu pego todas as palavras da sntenca mais importante
329         sent_imp_palavras = pipeline.tokenize(sentenca_mais_importante)
330         #aqui eu removo as palavras de parada
331         sent_imp_palavras = [plvr for plvr in sent_imp_palavras if plvr not in stops]
332
333         #aqui eu pego as tag associadas as palavras
334         tag_token = pipeline.tag(sent_imp_palavras)
335         entidades_nomeadas = []#esse é o array onde eu vou colocar as palavras identificadas como "N" ou "NPROP"
336         for token, tag in tag_token:
337             #aqui eu pego as palavras que possouem a tag de nome proprio
338             if tag == 'N' or tag == 'NPROP':
339                 entidades_nomeadas.append(token)
340
341         txt_final = ' '.join(entidades_nomeadas[:6])
342         self._print_console('finalizando extração de informacao')
343         self._bot_message('...')
344         return txt_final
```

Fonte: Elaborado pelo autor, 2021.

- **Obtenção do texto do usuário:** Esta é a etapa mais simples, consiste em pegar o texto da notícia fornecida inicialmente pelo usuário, sobre a qual será realizado o processo de análise. Esta etapa pode ser observada na linha 308, Figura 8.

- **Obtenção das palavras de parada:** Quando se trabalha com extração de informação de textos é necessário realizar uma etapa de limpeza extra, esta etapa consiste em remover palavras que conectam outras palavras, como: a, o, de, do, aquela, entre outras, junto das palavras de parada a serem removidas. Também pode-se remover a pontuação, pois elas podem ocasionar o mesmo problema, se a pontuação será incluída junta das palavras de parada depende da implementação. A remoção destas palavras é necessária pois elas não

agregam muito valor ao texto, e em uma contagem de quantas vezes uma palavra aparece, elas geralmente possuem as contagens mais altas, isso pode fazer com que o processo de ranqueamento de uma sentença se torne enviesado, pois pode acabar dando muito peso a palavras que não adicionam muito conteúdo ao texto. Esta etapa pode ser observada na linha 309, Figura 8.

- **Quebra do texto em sentenças:** Esta é uma etapa relativamente simples. Nela é pego o texto fornecido pelo usuário, que então é segmentado em várias sentenças que o compõem. Esta etapa é importante para ranqueamento mais à frente. Esta etapa pode ser observada na linha 310, Figura 8.

- **Quebra do texto em palavras:** Nesta etapa inicialmente, todas as palavras contidas no texto são convertidas para minúsculas, o texto então é segmentado pelas palavras que o compõem, estas palavras então são postas em um *array* para serem usadas posteriormente. Esta etapa pode ser observada na linha 311, Figura 8.

- **Remoção das palavras de parada:** Nesta etapa são removidas as palavras de parada do texto. Para remover-se estas palavras, é percorrido o *array* gerado anteriormente, onde é verificado se a palavra não se encontra no *array* das *stop-words*, caso se encontre a palavra em questão é ignorada. Esta etapa pode ser observada na linha 313 Figura 8, onde é usado um *for* para esta função.

- **Dicionário de frequência:** Com texto já divido em palavras, as palavras todas convertidas para minúsculas, e as palavras de para removidas, então é criado um dicionário da frequência destas palavras, onde a palavra se torna a chave deste dicionário e a quantidade de vezes que ela apareceu se torna seu valor. Esta etapa pode ser observada na linha 314, Figura 8.

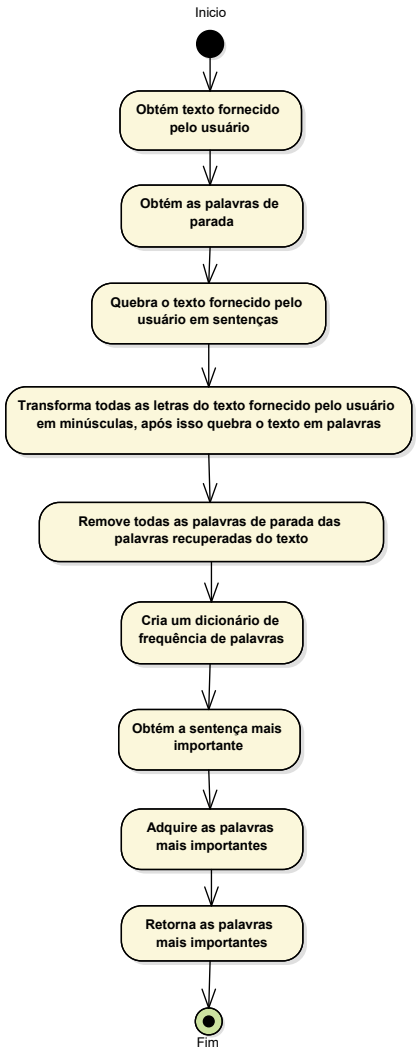
- **Obtenção da Sentença mais importante:** Para a realização desta etapa é necessário que as outras tenham sido concluídas. Para a realização desta etapa inicialmente é criado um *defaultdict* (é um dicionário em Python, a diferença é que caso seja fornecida uma chave que ele não contenha não é disparada uma exceção). Neste *defaultdict* a chave será o índice da sentença e seu valor será a pontuação total da sentença. Seguindo adiante, é inicializado um *for* que percorrerá todas as sentenças, dentro deste *for* é inicializado outro *for* que percorrerá todas as palavras da sentença, verificando se esta palavra se encontra no dicionário de frequência, caso se encontre o valor da palavra contido no dicionário de frequência é acrescentado a pontuação da sentença, para posterior ranqueamento. Esta etapa pode ser observada entre as linhas 317-323, Figura 8.

Após a atribuição de uma pontuação a cada sentença do texto, é adquirida a sentença de maior pontuação. A sentença mais importante então é dividida em palavras, tem as palavras de parada removidas, então obtém-se um *array* de palavras da sentença mais importante.

Para seguir para etapa final primeiro é necessário carregar o *tagger*. Para melhor entendimento sobre o *tagger*, voltar a seção 2.9.4.1. O NLTK não inclui um *tagger* para português, então é necessário treinar um para ser utilizado, eu utilizei um modelo de *tagger* já treinado (INOUE, 2019).

Com o *tagger* carregado, são percorridas as palavras da sentença mais importante, e recuperado os nomes próprios (NPROP) e os substantivos (N). Eles então são agrupados em uma única string que então é retornada.

Figura 9 – Fluxo de obtenção das palavras mais importantes



Fonte: Elaborado pelo autor, 2021.

3.4.2.2 Verificação de similaridade

Esta fase é executada após a finalização da execução do *spider*. O *spider* será melhor abordado na seção 3.4.3.

Esta etapa é iniciada após o Scrapy finalizar a aquisição das notícias com base nas palavras mais relevantes recuperadas.

Esta etapa poder ser segmentada em três partes: resumir as notícias obtidas, com as notícias resumidas verificar aquelas com a maior similaridade com a do usuário e fazer um tratamento nos dados retornados.

A verificação de similaridade será melhor detalhada nos tópicos apresentados a seguir.

- **Resumir notícias:** Os resumos das notícias são realizados utilizando a técnica descrita anteriormente para a atribuição de uma pontuação para a sentença, obtendo então aquela de maior valor. A diferença aqui está na quantidade de sentenças retornadas, essa quantidade é definida de acordo com o número de sentenças na notícia fornecida pelo usuário, para que o resumo tenha um tamanho semelhante ao da notícia, assim aumentando a similaridade que os textos podem ter.
- **Notícias com maior similaridade:** Com as notícias resumidas prontas, chega-se a esta etapa. Aqui verifica-se a similaridade do resumo com a notícia informada pelo usuário.
- **Tratamento de dados:** É removido qualquer *link* que esteja duplicado, onde dependendo da configuração também retorna o rótulo da notícia.

3.4.2.2.1 N-grama

Segundo Cavnar, Trenkle et al. (1994), n-grama é uma sequência de n caracteres dentro de um texto. Neste trabalho n-grama foi definido como sendo uma palavra.

3.4.2.2.2 Word Embeddings

Segundo Heidenreich (2018), pode ser definido como a representação vetorial de palavras reais. *Word Embeddings* são utilizadas, pois computadores não são capazes de compreender naturalmente uma palavra. Deste modo ao codificá-las em um vetor unidimensional, permite ao computador realizar mais facilmente procedimentos de análise, já que se torna mais viável a utilização de procedimentos matemáticos sobre elas.

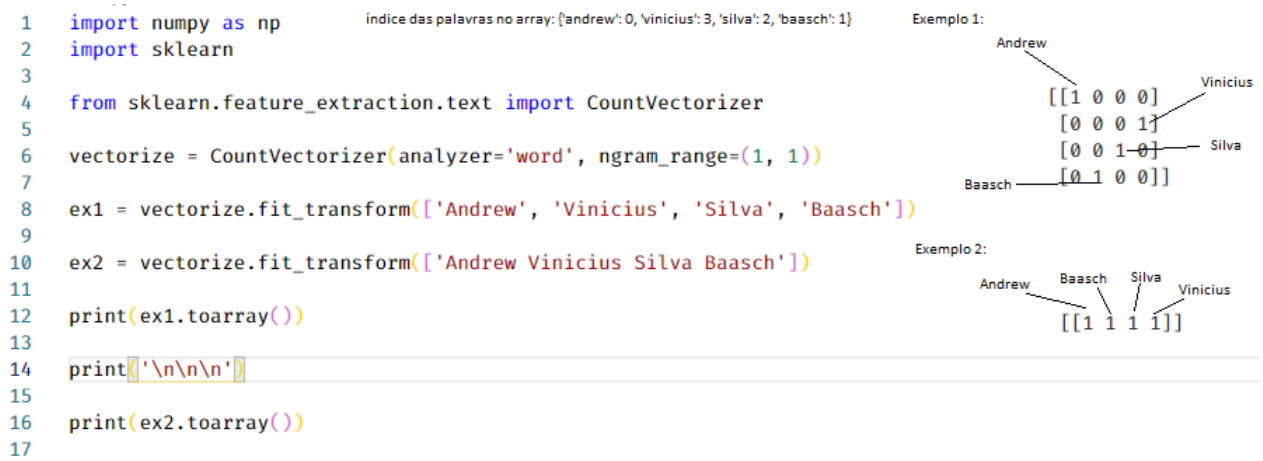
3.4.2.2.3 Hipótese de Distribuição

Hipótese originada no trabalho de Harris (1954), nele é levantada a hipótese de que palavras poderiam ser agrupadas em conjuntos que representam significados, onde as palavras contidas nestes conjuntos possuiriam significados semelhantes. Onde Heidenreich (2018), cita que, palavras que compartilham de um mesmo contexto tendem a possuir significados semelhantes, como palavras em uma mesma frase.

3.4.2.2.4 Vetorização de Contagem(Count Vectorizing)

É um método desenvolvido para a representação numérica de palavras, é dos métodos mais básicos desenvolvidos para este fim. A ideia base em sua utilização é relativamente simples. Inicialmente é criado um *array*, onde seu tamanho é correspondente a quantidade de palavras únicas contidas no texto informado. No *array* gerado, cada palavra única é representada por 1, os outros espaços do *array* são povoados por 0. É possível visualizar um exemplo na Figura 10.

Figura 10 – Exemplo do vetor de palavras gerado



Fonte: Elaborado pelo autor, 2021.

Na Figura 10 onde está escrito exemplo 1, é possível visualizar a representação do *array* com mais clareza, já que cada palavra foi posta em uma posição distinta do *array*. Na Figura 10, onde está escrito exemplo 2, ainda é a representação das palavras escolhidas, porém em um único *array*.

3.4.2.2.5 Cálculo de similaridade

Para a verificação de similaridade é utilizado o cálculo *Containment Similarity*, onde segundo Yang et al. (2019), a similaridade de contenção é obtida por meio da interseção de dois registros, X e Y, que é dividido pela quantidade de registros em X. Onde conforme mostra Broder et al. (1997), o resultado do cálculo resulta em um número entre 0 e 1, onde quanto mais próximo de 1, mais similares os registros são. Este cálculo surgiu levando em conta a grande quantidade de documentos disponíveis na internet, o que fez com houvesse uma grande proliferação de documentos quase idênticos (BRODER, 1997).

3.4.2.2.6 Algoritmo de verificação de similaridade

As linhas mencionadas a seguir podem ser visualizadas na Figura 11.

Para a execução do algoritmo de similaridade inicialmente é definida a quantidade de n-gramas que será utilizada, como mencionado anteriormente nesse trabalho o n-grama foi definido como sendo uma palavra, então o n-grama será definido como 1, que representa uma palavra, essa etapa pode ser visualizada na linha 9. Em seguida é instanciado o *ContVectorizer* que será utilizado para a criação da representação numérica das palavras, essa etapa pode ser visualizada na linha 10. Na linha 11, é criado o vetor de contagem das palavras. Na linha 13, é recuperado o vetor final, que foi gerado. Na linha 15, é calculada a interseção entre os vetores gerados para cada texto informado, onde será recuperado o valor mínimo da interseção dos dois textos, que será retornado em um novo vetor. Na linha 16 é recuperada a quantidade de n-gramas resultantes da interseção. Na linha 19, é recuperado a quantidade de n-gramas do vetor do texto de origem para verificar se ele se encontra no texto de destino. Na linha 22 é feito o cálculo que retornará o índice de similaridade, e por fim na linha 24 é retornado o índice de similaridade entre os textos.

O limite definido para a escolha das notícias similares foi definido como sendo 38%. Este número foi escolhido, pois com as notícias utilizadas para testes um valor maior não estava retornando notícias, mesmo elas sendo similares.

Figura 11 – Algoritmo de verificação de similaridade

```python
import numpy as np
import sklearn

from sklearn.feature_extraction.text import CountVectorizer


def verifica_similaridade(txt1, txt2):
    #Numero de n-gram
    n = 1
    counts = CountVectorizer(analyzer='word', ngram_range=(n, n))
    n_grams = counts.fit_transform([txt1, txt2])

    n_grams_array = n_grams.toarray()

    intersection_list = np.amin(n_grams.toarray(), axis = 0)
    intersection_count = np.sum(intersection_list)

    index_A = 0
    A_count = np.sum(n_grams.toarray()[index_A])

    #valor de similaridade
    val_sim = intersection_count/A_count

    return val_sim
```

Fonte: Elaborado pelo autor, 2021.

3.4.3 Estrutura do Scrapy

Módulo responsável por buscar as notícias. As notícias são obtidas por meio do G1. Inicialmente é fornecido as palavras para a realização da pesquisa, em seguida são definidos os *spiders* a serem executados, então é iniciado o processamento que executa os *spiders*. Para a execução dos *spiders* é necessário criar um novo processo para a execução do Scrapy, isso é necessário pois o Scrapy utiliza uma biblioteca chamada twisted, que serve para gerenciar algumas requisições web. O problema é que o twisted não pode ser reinicializado e precisa ser executa no thread principal, porém na thread principal já está sendo executado o telegram-python-bot, e para resolver este problema foi utilizado o multiprocessamento.

Ao finalizar a aquisição de notícias, o Scrapy foi programado por mim, para salvar o resultado da busca em json, este json é lido posteriormente para a recuperação de seu conteúdo, que segue para a etapa do PLN que então retorna o resultado para o controller, que por sua vez mostra o resultado para o usuário.

3.4.4 Modelo de detecção

No decorrer do desenvolvimento do trabalho, foi desenvolvido um modelo de detecção em *fake news* em Python. No entanto no decorrer dos testes verificou-se que modelo estava apontando a grande maioria das notícias como falsas. Nos testes também foi observado que no-

tícias mais longas tendiam a ser classificadas como verdadeiras, e notícias mais curtas tendiam mais a serem classificadas como falsas, e como pode ser observado no trabalho de Pérez-Rosas et al. (2017), estilo de escrita, pontuação utilizada, entre outros fatores influenciam neste tipo de classificação, no trabalho de Guarise (2019), também foram observadas tendencias semelhantes. Para contornar este problema foi limitada a quantidade de caracteres, possíveis de serem informadas, porém isso não resolveu o problema. Para contornar esta situação seriam necessários mais testes, e um estudo mais aprofundado nos métodos de treinamento do modelo, porém não houve tempo hábil para executar esses passos, por este motivo o modelo desenvolvido veio a ser descartado. Os dados utilizados para o treinamento foram obtidos de um corpus em português do Brasil (MONTEIRO et al., 2018b).

4 RESULTADOS E DISCUSSÕES

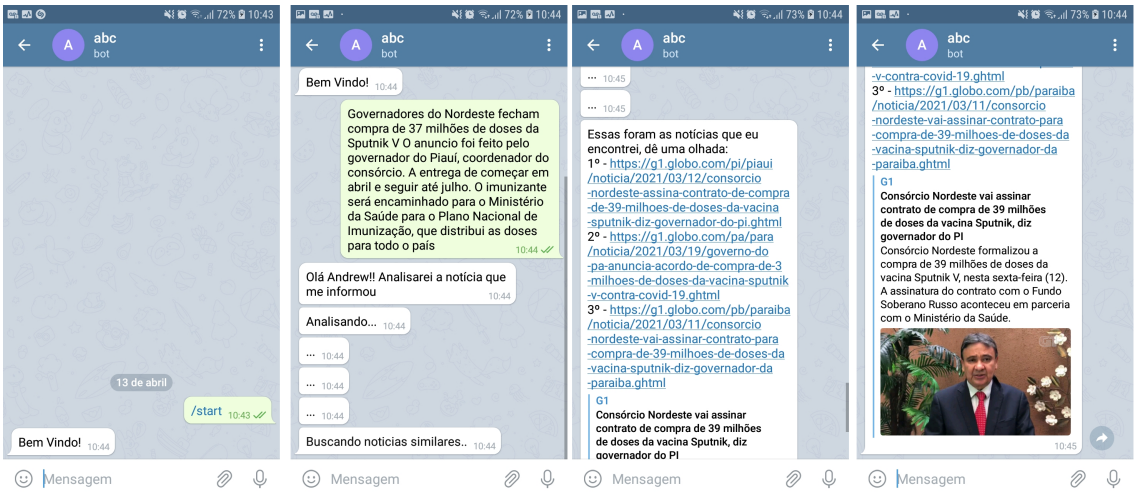
Nesta seção serão apresentados os resultados obtidos, e as lições aprendidas durante o processo de desenvolvimento.

O desenvolvimento do bot utilizando o Telegram é relativamente simples, pois a API do Telegram é muito bem documentada, e a equipe de desenvolvimento da API em Python disponibiliza vários exemplos, sendo que a API em Python é a API do Telegram, apenas convertida para Python, pois a linguagem nativa da API é outra. O desenvolvimento de um sistema de coleta dados e posterior armazenamento, também é relativamente simples utilizando Scrapy.

4.1 TELAS DO SISTEMA

Na Figura 12, é possível visualizar as telas do sistema. Da esquerda para a direta se encontra a tela de boas-vindas, apresentada ao usuário no primeiro acesso, em seguida a tela que aparece quando o usuário informa uma notícia, onde o sistema saúda o usuário e mostra as etapas que está desempenhando, na próxima tela são apresentados os links das notícias similares, na última tela é apresentada uma previa de uma das notícias apresentadas.

Figura 12 – Algoritmo de verificação de similaridade



Fonte: Elaborado pelo autor, 2021.

4.2 OBSTÁCULOS ENCONTRADOS

Uma dificuldade encontrada no decorrer deste projeto, foi a definição das tecnologias a serem utilizadas, partindo desde a escolha da linguagem a ser utilizada, até a ferramenta utilizada para a interação com usuário. Definir essas tecnologias é difícil. No início se segue

por linha que se acredita que dará resultado, para que no fim se descubra que ela não será viável, pois acabou-se por descobrir que a ferramenta é muito complexa ou pesada para execução do que se deseja, o que inviabiliza sua utilização no projeto final.

A maior dificuldade no processo de desenvolvimento deste projeto, foi unir diferentes tecnologias que trabalham com *thread*, pois tanto a API em Python do Telegram quanto o Scrapy, necessitam ser executadas na *thread* principal, o que ocasiona conflitos, com erros de execução, e mensagens informando que determinada ferramenta só pode ser utilizada na *thread* principal. A resolução deste problema levou tempo, sendo cogitada a hipótese de descartar o que havia sido desenvolvido até o momento, e partir em busca de outra solução. Esta possibilidade foi considerada, pois inicialmente acreditava-se que para a união das tecnologias seria necessário compreender profundamente o funcionamento de cada uma delas, e desenvolver uma única aplicação que as tratasse como uma coisa só. O projeto inicial foi mantido pois foi encontrada uma solução que possibilita a execução de ambas as ferramentas, sem que uma entrasse em conflito com a outra. Isso foi possível, pois as tecnologias são executadas em processos distintos, o que possibilitou sua utilização em conjunto.

A abordagem inicial do trabalho visava, vasculhar a internet em busca da fonte original de determinada notícia, ou o mais próximo disto. No entanto, percebeu-se que o maior empecilho para a realização desta abordagem estava no fato de que as informações contidas na internet, dificilmente estão bem estruturadas, e caso estejam, esta estrutura não é a mesma para todos os sites, pois não há um padrão de estruturação na rede. Ainda foi insistido mais um pouco nesta abordagem, onde acabou-se por encontrar um ramo de pesquisa que visa estruturar as informações contidas na internet, este ramo é chamado de Web Semântica, porém acabou-se por confirmar que esta abordagem, não seria viável, e o estudo relacionado a Web Semântica resultaria em outro TCC.

4.3 LIÇÕES APRENDIDAS

Possuir entusiasmo é bom, porém já sair começado a desenvolver algo sem antes ter avaliado os outros componentes do projeto, ou mesmo não ter levado em conta a escassez de conteúdos em seu idioma, pode ser um erro crítico, levando ao descarte de uma parte inteira do projeto. É necessário avaliar com cuidado as opções e recursos disponíveis.

Percebeu-se também a necessidade da realização de uma pesquisa mais profunda sobre as ferramentas, se concentrando em seu mecanismo de funcionamento interno, e como isso pode acabar afetando o projeto como um todo, ao ser mesclado com outras tecnologias. O gerenciamento do tempo e planejamento das atividades a serem executas é algo fundamental, sem isso o andamento do projeto pode vir a ser muito afetado.

4.4 TRABALHOS CORRELATOS

Aqui são mostrados alguns trabalhos que possuem relação com o software desenvolvido, bem como a abordagem que foi utilizada em sua elaboração.

Todos os trabalhos aqui citados têm como objetivo desenvolver um meio de automatiza a detecção de *fake news*, tendo em vista o potencial nocivo que podem vir a ter. Foi observado que todos os trabalhos desenvolveram um modelo de análise, baseado em algum tipo de algoritmo de aprendizado de máquina, porém nenhum deles procura em sites confiáveis a notícia informada, para encontrar notícias similares, o que é o diferencial deste trabalho.

4.4.1 Detecção de notícias falsas usando técnicas de deep learning

O Guarise (2019), foi desenvolvido tendo em vista a enxurrada de *fake news* que se tornou mais intensa a partir de 2016, observando que no Brasil em 2018 também foi marcada fortemente pela presença de *fake news*.

Este trabalho ainda tem como objetivo desenvolver uma solução automatizada que visa acelerar o processo de detecção de *fake news*, já que para um humano analisar notícia por notícia manualmente se mostra inviável (GUARISE, 2019).

O desenvolvimento do trabalho foi feito em cima da hipótese de que notícias jornalísticas focam-se nos fatos, enquanto *fake news* apela mais para o lado emocional. A partir disto são usados algoritmos de aprendizado de máquina que tenha a capacidade de encontrar diferenças nos padrões de escrita de uma notícia falsa e uma verdadeira (GUARISE, 2019). .

O treinamento utilizado neste modelo foi feito utilizando uma base de dados de notícias falsas e verdadeiras em português do Brasil e como saída o modelo gerado informa se a notícia que foi repassada a ele é falsa ou verdadeira (GUARISE, 2019).

Este trabalho e o de Guarise, propõem uma ferramenta de auxílio na identificação de *fake news*. Sendo que no trabalho proposto por (GUARISE, 2019), pretende-se futuramente a disponibilização de aplicativo que seja acessível a população, enquanto no sistema desenvolvido este foi um dos requisitos principais. Podendo ser observado na Tabela 1, a similaridade entre os sistemas.

Tabela 1 – Tabela de similaridade entre APLNATINS - Guarise

Recursos	APLNATINS	Trabalho Correlato
Pré processamento de texto	V	V
Sumarização de Texto	V	V
Análise de Texto	V	V
Modelo Classificação	F	V
Obtenção de Notícias Similares	V	F
Acessível a usuário comum	V	F

Fonte: Elaborado pelo autor, 2021.

4.4.2 Deep Learning para Classificação de *fake news* por Sumarização de Texto

Este trabalho foi desenvolvido tendo-se observado que houve uma proliferação de *fake news*, que sua disseminação pode afetar as pessoas das mais diversas formas, e que a análise manual dessas notícias tende a ser custosa e não capaz de acompanhar a difusão deste tipo de notícia (MARUMO, 2018).

Para solucionar este empecilho Marumo desenvolveu um modelo de análise de texto utilizando de aprendizado de máquina, que possibilita a classificação de texto de modo rápido, que pode vir a ser melhor que o ser humano (MARUMO, 2018). O princípio de funcionamento do sistema, consiste em sumarizar uma notícia informada, para posteriormente fornecê-la ao modelo de classificação treinado, que por sua vez fornece um rótulo para a notícia sendo, falsa ou verdadeira. Focou-se na utilização de diferentes técnicas de aprendizado de máquina para criação do modelo

No desenvolvimento do modelo foram utilizadas algumas técnicas que envolvera a utilização de palavras-chave populares que são utilizadas nos títulos e conteúdos destes textos, já que foi percebido que as *fake news* possuem um padrão em sua utilização. Foi utilizada também a sumarização destes textos, o que poderia auxiliar em sua classificação (MARUMO, 2018).

Para treinar este modelo foi utilizada uma grande quantidade de dados em formato de texto, e a partir deles foi foram utilizadas técnicas de análise e extração de conhecimento, que foram utilizados a fim de encontrar padrões que conseguissem classificar uma notícia como falsa ou não (MARUMO, 2018). O modelo mostra como saída se a notícia informada é falsa ou não de acordo com a análise do sistema.

Assim como no trabalho de MARUMO (2018), o sistema desenvolvido também inclui etapas de pré-processamento de texto para posterior análise, como também inclui uma etapa de sumarização para verificação da similaridade das notícias recuperadas com a notícia informada pelo usuário. A principal diferença está na utilização de um modelo de classificação para a

categorização da notícia, onde ele chegou a ser desenvolvido, porém não foi utilizado na versão final do sistema. A Tabela 2, mostra os recursos, existentes em ambos os trabalhos.

Tabela 2 – Tabela de similaridade entre APLNATINS - Marumo

Recursos	APLNATINS	Trabalho Correlato
Pré processamento de texto	V	V
Sumarização de Texto	V	V
Análise de Texto	V	V
Modelo Classificação	F	V
Obtenção de Notícias Similares	V	F
Acessível a usuário comum	V	F

Fonte: Elaborado pelo autor, 2021.

4.4.3 Contribuições para o estudo de notícias falsas em Português

Observando o imenso estrago ocasionado pela propagação de *fake news*, e a ausência de dados previamente rotulados em português do Brasil, para a utilização em modelos de classificação. Este desenvolveu um banco de dados para armazenamento de notícias já corretamente rotuladas, e o disponibilizou para o público. O presente trabalho também disponibilizou um bot, acessível pelo WhatApp, não se pode informar uma notícia e aguardar a avaliação do bot (MONTEIRO et al., 2018a).

Ambos os sistemas possuem a proposta de auxiliar o usuário na detecção de *fake news*, sendo o trabalho (MONTEIRO et al., 2018a), focado mais na utilização de um modelo de classificação para este fim. Na Tabela 3, pode-se observar a semelhança entre as soluções.

Tabela 3 – Tabela de similaridade entre APLNATINS - Monteiro

Recursos	APLNATINS	Trabalho Correlato
Pré processamento de texto	V	V
Sumarização de Texto	V	F
Análise de Texto	V	V
Modelo Classificação	F	V
Obtenção de Notícias Similares	V	F
Acessível a usuário comum	V	V

Fonte: Elaborado pelo autor, 2021.

5 CONSIDERAÇÕES FINAIS

Este trabalho desenvolveu um sistema que pode servir de auxílio na identificação de *fake news*, tendo em vista o quão difícil pode ser discernir o que pode vir a ser um fato, daquilo que é ficção. Como pôde ser observado no início do cápitulo 2, Infodemia, Pós Verdade, Bolha dos Filtros e Viés de confirmação, são fatores que colaboram para o caos que as *fake news* podem ocasionar, onde o problema mais grave provocado pela disseminação de *fake news*, sendo aquele que envolve saúde pública, com pessoas negando a eficácia da vacinação.

É importante observar a relevância das agências de checagem de fatos, pois neste momento em que o mundo se encontra, desempenham um papel vital, procurando informar e instruir a população em relação as notícias falsas.

O sistema foi desenvolvido de modo a prover fácil acesso as pessoas, pois utiliza de um aplicativo de comunicação muito popular exigindo do usuário apenas a instalação do aplicativo, utilizar a busca fornecida de dentro do aplicativo e procurar pelo nome do bot, na versão atual o bot chama-se "abc"será alterado posteriormente para "APLNATINS". O sistema possui as seguintes capacidades:

- Interação facilitada não requerendo conhecimentos específicos por parte do usuário. Bastando apernas enviar a notícia para o bot.
- Possui uma interface amigável, e de fácil entendimento.
- Identifica a sentença mais importante no texto da notícia informada.
- Fornece ao usuário notícias similares com aquela informada, possibilitando a ele uma análise mais criteriosa quanto a veracidade da notícia.
- Caso não encontre notícias similares indica ferramentas de suporte alternativas, para que o usuário não fique perdido.

Para a identificação de *fake news*, as formas mais tradicionais consistem em levantar todo o conteúdo relacionado a notícia em questão, com esses dados obtidos identificar informações que corroborem ou refutem a notícia em questão, e ainda pode-se recorrer ao auxílio de especialistas no assunto para se chegar a um veredito (LUPA, 2015a). Outra abordagem utilizada consiste em consultar quem divulgou a informação, checar fontes de origem confiável, verificar fontes oficiais e caso necessário consultar fontes alternativas que podem contrariar os fatos oficiais (FATOS, 2021a). A grande limitação dessas abordagens é que elas consistem em um

trabalho bastante manual, que não consegue acompanhar a proliferação de *fake news*. O trabalho desenvolvido oferece uma solução mais automatizada, que pode ser utilizada em conjunto com as técnicas já existentes, potencializando a área de atuação.

O bot desenvolvido complementa as formas automáticas de detecção, fornecendo aos modelos de classificação automática notícias similares com aquela informada, com a vantagem de já terem sido checadas podendo servir de auxílio no momento da classificação da notícia como sendo falsa ou verdadeira.

O código da versão atual do projeto pode ser acessado no meu git (BAASCH, 2021). Na versão atual o bot pode ser encontrado no Telegram pelo nome "abc"ou "@BotAbc123_bot". Na primeira execução o bot pode demorar a responder, pois durante um período de inatividade o servidor "dorme", até que ele "acorde", pode levar 10min.

Durante o desenvolvimento, inicialmente foram definidos os requisitos que o sistema deveria atingir. Com os requisitos principais levantados, foi elaborado o fluxo de execução básico que o sistema deveria executar. Após a elaboração deste esboço, foi desenvolvido o diagrama de fluxo, que pode ser observado na Figura 7, onde foi detalhado o processo de execução a ser executado pelo sistema. Com os requisitos iniciais definidos e o diagrama de fluxo finalizado, foi feita uma pesquisa em busca das possíveis ferramentas que poderiam ser utilizadas para a implementação destes requisitos. Assim é possível perceber que ao decorrer do desenvolvimento do projeto, foi seguido as práticas da Engenharia de *Software*, análise, planejamento e execução.

Com o sistema desenvolvido acredita-se, que possa vir a servir de auxílio na batalha contra as *fake news*, dado ao usuário a possibilidade de averiguar a existências de notícias minimamente parecidas com aquela informada.

É importante salientar, que algumas das bibliotecas utilizadas no desenvolvimento deste trabalho, podem não existir mais daqui a 10 anos.

5.1 TRABALHOS FUTUROS

O sistema desenvolvido possui uma interação com usuário não muito explorada, sendo possível a interação somente através de texto, não sendo possível analisar áudio, vídeo ou imagem. Sendo uma funcionalidade que pode vir a ser adicionada no futuro. A API do Telegram viabiliza diversas funcionalidades que não foram exploradas, pode-se estudar mais a fundo a API, e desenvolver um meio de interação mais robusto.

O sistema desenvolvido aplica as técnicas apresentadas na seção 2.7, porém de modo simplificado, as técnicas utilizadas podem ser aprimoradas para um melhor aproveitamento do sistema. Algumas das etapas que podem ser aprimoradas serão mencionadas adiante.

A parte que cuida do processo de análise do texto pode ser melhorada, por meio da inclusão de técnicas de extração de informação. A extração de informação pode incluir as etapas de análise sintática e semântica, a extração de informação pode utilizar de modelos de análise de dados treinados. Ainda para este mesmo propósito também podem ser desenvolvidas regras baseadas em expressões regulares. Ainda na etapa de extração de informação podem ser aplicadas técnicas de PLN mais avanças como a utilização de árvores, que mostrem a relação de dependência das palavras em uma estrutura de árvores. Ainda nesta etapa podem ser aplicadas outras técnicas de extração mais avançada, estes processos citados estariam contidos na etapa de análise sintática. Poderiam ainda ser aplicadas técnicas que visem compreender o significado das palavras levando em conta seu contexto. Esta etapa se enquadraria na análise semântica.

Futuramente também pretende-se desenvolver uma versão do sistema que opere por meio do *WhatsApp*.

O bot desenvolvido pode ser adaptado para ser utilizado em outros contextos, além da identificação de *fake news*. Podendo ser alterado para auxiliar o usuário a adquirir mais conhecimento em uma área em que ele venha possuir um maior interesse, ou para desempenhar outras atividades.

REFERÊNCIAS

AIZAWA, A. An information-theoretic perspective of tf–idf measures. **Information Processing & Management**, Elsevier, v. 39, n. 1, p. 45–65, 2003.

ALLEN, J. **Natural language understanding**. [S.l.]: Benjamin-Cummings Publishing Co., Inc., 1995.

ALMEIDA, A. S. de. As bibliotecas universitárias no combate à infodemia. **RevIU. Revista Informação & Universidade**, v. 2, p. 1–19, 2020.

AMARAL, D. O. F. d. **O reconhecimento de entidades nomeadas por meio de conditional random fields para a língua portuguesa**. Dissertação (Mestrado) — Pontifícia Universidade Católica do Rio Grande do Sul, 2013.

ARROYO-SÁNCHEZ, A. S.; CABREJO, P.; CRUZADO, V. Infodemia, la otra pandemia durante la enfermedad por coronavirus 2019. **An Fac Med**, v. 81, n. 2, p. 2, 2020.

BAASCH, A. V. da S. **Aplicaçã de Processamento de Linguagem Natural para Análise de Texto e Indicaçã de Notícias Similares**. 2021. Disponível em: <https://github.com/Volpe6/TelegramBotPython>. Acesso em: 3 abr. 2021.

BBC. **Como o termo 'fake news' virou arma nos dois lados da batalha política mundial**. 2018. Disponível em: <https://www.bbc.com/portuguese/internacional-42779796>. Acesso em: 01 mai. 2020.

BBC. **'Ganhava a vida escrevendo notícias falsas'**. 2019. Disponível em: <https://www.bbc.com/portuguese/vert-fut-48832474>. Acesso em: 06 mai. 2020.

BBC. **How liars create the 'illusion of truth'**. 2020. Disponível em: <https://www.bbc.com/future/article/20161026-how-liars-create-the-illusion-of-truth>. Acesso em: 21 set. 2020.

BIRD, S.; KLEIN, E.; LOPER, E. **Natural language processing with Python: analyzing text with the natural language toolkit**. [S.l.]: "O'Reilly Media, Inc.", 2009.

BRODER, A. Z. On the resemblance and containment of documents. In: IEEE. **Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)**. [S.l.], 1997. p. 21–29.

BRODER, A. Z.; GLASSMAN, S. C.; MANASSE, M. S.; ZWEIG, G. Syntactic clustering of the web. **Computer networks and ISDN systems**, Elsevier, v. 29, n. 8-13, p. 1157–1166, 1997.

CAVNAR, W. B.; TRENKLE, J. M. et al. N-gram-based text categorization. In: CITESEER. **Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval**. [S.l.], 1994. v. 161175.

CHOWDHURY, G. G. Natural language processing. **Annual review of information science and technology**, Wiley Online Library, v. 37, n. 1, p. 51–89, 2003.

DICIONÁRIO DE OXFORD. **post-truth**. 2021. Disponível em: <https://www.oxfordlearnersdictionaries.com/definition/english/post-truth>. Acesso em: 07 feb. 2021.

DICIONÁRIO PRIBERAM DA LÍNGUA PORTUGUESA. **Dicionário Priberam da Língua Portuguesa**. 2020. Disponível em: <https://dicionario.priberam.org>. Acesso em: 05 abr. 2020.

DORETTO, J. "minhas próprias notícias": jornalismo e o público jovem brasileiro e português em contexto digital. **Intercom: Revista Brasileira de Ciências da Comunicação**, SciELO Brasil, v. 42, n. 1, p. 113–129, 2019.

FABER, J. Viés cognitivo: Quando ser racional não é o bastante/cognitive bias: When being rational is not enough. **Health Sciences Journal**, v. 4, n. 4, p. 2–8, 2014.

FACEBOOK. **FastText Library for efficient text classification and representation learning**. 2020. Disponível em: <https://fasttext.cc/>. Acesso em: 3 abr. 2021.

FATOS, A. **Nosso Método**. 2021. Disponível em: <https://www.aosfatos.org/nosso-m\\%5B\\%C3\\%A9e\\%5Dtodo/>. Acesso em: 21 abr. 2021.

FATOS, A. **O que é checagem de fatos — ou fact-checking?** 2021. Disponível em: <https://www.aosfatos.org/checagem-de-fatos-ou-fact-checking/>. Acesso em: 21 abr. 2021.

FAUSTINO, A. **Fake News: A Liberdade de Expressão nas Redes Sociais na Sociedade da Informação**. [S.l.]: Lura Editorial, 2020.

FERREIRA, J. D. C. **Python para Pré-processamento e Extracção de Características a partir de Texto Português**. Tese (Doutorado) — Universidade de Coimbra, 2019.

FERREIRA, R. R. Rede de mentiras: a propagação de fake news na pré-campanha presidencial brasileira. **Observatorio (OBS*)**, v. 12, n. 5, 2018.

FLETCHER, R.; NIELSEN, R. K. Are news audiences increasingly fragmented? a cross-national comparative analysis of cross-platform news audience fragmentation and duplication. **Journal of Communication**, Oxford University Press, v. 67, n. 4, p. 476–498, 2017.

FREITAS, F.; SCHULZ, S. Ontologias, web semântica e saúde. **Revista Eletrônica de Comunicação, Informação e Inovação em Saúde**, v. 3, n. 1, 2009.

G1. **Três anos depois, linchamento de Fabiane após boato na web pode ajudar a endurecer lei**. 2017. Disponível em: <https://g1.globo.com/e-ou-nao-e/noticia/tres-anos-depois-linchamento-de-fabiane-apos-boato-na-web-pode-ajudar-a-endurecer-lei.ghtml>. Acesso em: 07 feb. 2021.

G1. **G1 lança Fato ou Fake, novo serviço de checagem de conteúdos suspeitos**. 2018. Disponível em: <https://g1.globo.com/fato-ou-fake/noticia/2018/07/30/g1-lanca-fato-ou-fake-novo-servico-de-checagem-de-conteudos-suspeitos.ghtml>. Acesso em: 21 abr. 2021.

GEHLEN, M. A. Fact-checking: o caso da lupa, a primeira agência de checagem de notícias do brasil. **FRONTEIRAS DO JORNALISMO E MODELOS DE NE GÓCIO**, p. 44, 2011.

GELFERT, A. Fake news: A definition. **Informal Logic**, Informal Logic, v. 38, n. 1, p. 84–117, 2018.

GOOGLE BLOG. **Personalized Search for everyone**. 2010. Disponível em: <https://googleblog.blogspot.com/2009/12/personalized-search-for-everyone.html>. Acesso em: 18 mai. 2020.

GRUBER, T. **What is an Ontology?** 1993. Disponível em: <http://www.ksl.stanford.edu/kst/what-is-an-ontology.html>. Acesso em: 25 mar. 2021.

GUARISE, L. Detecção de notícias falsas usando técnicas de deep learning. 2019.

HARRIS, Z. S. Distributional structure. **Word**, Taylor & Francis, v. 10, n. 2-3, p. 146–162, 1954.

HEIDENREICH. **Introduction to Word Embeddings**. 2018. Disponível em: <https://towardsdatascience.com/introduction-to-word-embeddings-4cf857b12edc>. Acesso em: 20 abr. 2021.

HEROKU. **Heroku**. 2021. Disponível em: <https://www.heroku.com/what>. Acesso em: 29 mar. 2021.

INDURKHYA, N.; DAMERAU, F. J. **Handbook of natural language processing**. [S.l.]: CRC Press, 2010. v. 2.

INOUE, M. **POS-tagger-portuguese-nltk**. 2019. Disponível em: <https://github.com/inoueMashuu/POS-tagger-portuguese-nltk>. Acesso em: 29 mar. 2021.

ITAGIBA, G. Fake news e internet: esquemas, bots e a disputa pela atenção. **ITS**, 2017.

JUNIOR, G. C. Pós-verdade: a nova guerra contra os fatos em tempos de fake news. **ETD: Educaçao Temática Digital**, Faculdade de Educaçao, v. 21, n. 1, p. 278–284, 2019.

LUPA. **Como a Lupa faz suas checagens?** 2015. Disponível em: <https://piaui.folha.uol.com.br/lupa/2015/10/15/como-fazemos-nossas-checagens/>. Acesso em: 21 abr. 2021.

LUPA. **O que é a Agência Lupa?** 2015. Disponível em: <https://piaui.folha.uol.com.br/lupa/2015/10/15/como-selecionamos-as-frases-que-serao-checadas/>. Acesso em: 21 abr. 2021.

MARTIN, A. Digital literacy and the "digital society". **Digital literacies: Concepts, policies and practices**, Peter Lang New York, NY, v. 30, n. 2008, p. 151–176, 2008.

MARUMO, F. S. **Deep Learning para classificação de Fake News por sumarização de texto**. [S.l.]: Londrina, 2018.

MCINTYRE, L. **Post-truth**. [S.l.]: MIt Press, 2018.

MERRIAM-WEBSTER. **The Real Story of 'Fake News'**. 2020. Disponível em: <https://www.merriam-webster.com/words-at-play/the-real-story-of-fake-news>. Acesso em: 01 mai. 2020.

MITCHELL, R. **Web scraping with Python: Collecting more data from the modern web**. [S.l.]: "O'Reilly Media, Inc.", 2018.

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

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.

MÜLLER, A. C.; GUIDO, S. **Introduction to machine learning with Python: a guide for data scientists**. [S.l.]: "O'Reilly Media, Inc.", 2016.

OMS. **Managing epidemics: key facts about major deadly diseases**. 2018. Disponível em: <https://apps.who.int/iris/handle/10665/272442>. Acesso em: 07 feb. 2021.

PARISER, E. **O filtro invisível: o que a internet está escondendo de você**. [S.l.]: Editora Schwarcz-Companhia das Letras, 2012.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V. et al. Scikit-learn: Machine learning in python. **the Journal of machine Learning research**, JMLR. org, v. 12, p. 2825–2830, 2011.

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

PRIMO, A. F. T. Transformações no jornalismo em rede: sobre pessoas comuns, jornalistas e organizações; blogs, twitter, facebook e flipboard. **Intexto**, n. 25, p. 144–161, 2011.

PSYCHOLOGY TODAY. **When Correcting a Lie, Don't Repeat It. Do This Instead**. 2018. Disponível em: <https://www.psychologytoday.com/us/blog/words-matter/201807/when-correcting-lie-dont-repeat-it-do-instead-2>. Acesso em: 21 set. 2020.

PYTHON. **Estruturas de dados**. 2021. Disponível em: <https://docs.python.org/pt-br/3/tutorial/datastructures.html>. Acesso em: 3 abr. 2021.

PYTHONTELEGRAMBOT. **python-telegram-bot**. 2021. Disponível em: <https://github.com/python-telegram-bot/python-telegram-bot>. Acesso em: 3 abr. 2021.

RAMOS, J. et al. Using tf-idf to determine word relevance in document queries. In: CITESEER. **Proceedings of the first instructional conference on machine learning**. [S.l.], 2003. v. 242, n. 1, p. 29–48.

RAUTENBERG, S.; CARMO, P. R. V. do. Big data e ciência de dados. **Brazilian Journal of Information Science: research trends**, v. 13, n. 1, p. 56–67, 2019.

RESENDE, G.; MELO, P.; REIS, J. C.; VASCONCELOS, M.; ALMEIDA, J. M.; BENEVENUTO, F. Analyzing textual (mis) information shared in whatsapp groups. In: **Proceedings of the 10th ACM Conference on Web Science**. [S.l.: s.n.], 2019. p. 225–234.

RYDNING, D. R.-J. G.-J. The digitization of the world from edge to core. **Framingham: International Data Corporation**, 2018.

SANTOS, D. M. dos; TEIXEIRA, W. M. Fake news: a experiência de fatos em contexto de proliferação de informações falsas. 2019.

SANTOS, R.; AZEVEDO, J.; PEDRO, L. Literacia (s) digital (ais): definições, perspetivas e desafios. **Media & Jornalismo**, v. 15, n. 27, p. 17–44, 2015.

SARAIVA, L. J.; FARIA, J. F. D. A ciência e a mídia: A propagação de fake news e sua relação com o movimento anti-vacina no brasil. **Sociedade Brasileira de Estudos Interdisciplinares da Comunicação. Pará**, v. 42, n. 01, p. 01–15, 2019.

SASTRE, A.; OLIVEIRA, C. S. P. de; BELDA, F. R. A influência do "filtro bolha" na difusão de fake news nas mídias sociais: reflexões sobre as mudanças nos algoritmos do facebook. **Revista GEMInIS**, v. 9, n. 1, p. 4–17, 2018.

SCRAPY. **Scrapy**. 2020. Disponível em: <https://docs.scrapy.org/en/latest/topics/architecture.html>. Acesso em: 29 mar. 2021.

SELENIUM. **The Selenium Browser Automation Project**. 2021. Disponível em: <https://www.selenium.dev/documentation/en/>. Acesso em: 21 abr. 2021.

SPACY. **spaCy 101: Everything you need to know**. 2021. Disponível em: <https://spacy.io/usage/spacy-101>. Acesso em: 25 mar. 2021.

STANFORD. **Stemming and lemmatization**. 2008. Disponível em: <https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html>. Acesso em: 25 mar. 2021.

TALWAR, S.; DHIR, A.; KAUR, P.; ZAFAR, N.; ALRASHEEDY, M. Why do people share fake news? associations between the dark side of social media use and fake news sharing behavior. **Journal of Retailing and Consumer Services**, Elsevier, v. 51, p. 72–82, 2019.

TELEGRAM. **Telegram**. 2021. Disponível em: <https://telegram.org/>. Acesso em: 29 mar. 2021.

THE WASHINGTON POST. **When the Buzz Bites Back**. 2003. Disponível em: <https://www.washingtonpost.com/archive/opinions/2003/05/11/when-the-buzz-bites-back/bc8cd84f-cab6-4648-bf58-0277261af6cd/>. Acesso em: 07 feb. 2021.

TROTTA, R. **Os riscos das notícias falsas sobre saúde**. 2020. Disponível em: <https://medicinasa.com.br/fake-news-saude/>. Acesso em: 21 abr. 2021.

YANG, Y.; ZHANG, Y.; ZHANG, W.; HUANG, Z. Gb-kmv: An augmented kmv sketch for approximate containment similarity search. In: IEEE. **2019 IEEE 35th International Conference on Data Engineering (ICDE)**. [S.l.], 2019. p. 458–469.