UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

PEDRO LAMKOWSKI DOS SANTOS

INTELIGÊNCIA ARTIFICIAL EXPLICÁVEL APLICADA À CLASSIFICAÇÃO DE DESINFORMAÇÃO

BAURU
Dezembro/2020

PEDRO LAMKOWSKI DOS SANTOS

INTELIGÊNCIA ARTIFICIAL EXPLICÁVEL APLICADA À CLASSIFICAÇÃO DE DESINFORMAÇÃO

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Orientador: Prof. Dr. João Paulo Papa
Coorientador: Gustavo Henrique de Rosa

BAURU
Dezembro/2020

Pedro Lamkowski dos Santos Inteligência Artificial Explicável Aplicada à Classificação de Desinformação/ Pedro Lamkowski dos Santos. – Bauru, Dezembro/2020- 48 p. : il. (algumas color.) ; 30 cm.
Orientador: Prof. Dr. João Paulo Papa
Trabalho de Conclusão de Curso – Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Ciência da Computação, Dezembro/2020.
1. Processamento de Linguagem Natural 2. Aprendizado de Máquina 3. Inteligência Artificial Explicável 4. Detecção de Desinformação 5. Detecção de Fake News

.

Pedro Lamkowski dos Santos

Inteligência Artificial Explicável Aplicada à Classificação de Desinformação

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. João Paulo Papa
Orientador
Departamento de Computação
Faculdade de Ciências
Universidade Estadual Paulista "Júlio de Mesquita Filho"

Profa. Dra. Simone das Graças Domingues Prado
Departamento de Computação
Faculdade de Ciências
Universidade Estadual Paulista "Júlio de Mesquita Filho"

Prof. Dr. João Eduardo Machado Perea Martins
Departamento de Computação
Faculdade de Ciências
Universidade Estadual Paulista "Júlio de Mesquita Filho"

Bauru, 16 de Dezembro de 2020.

Dedico este trabalho àqueles que contribuíram e contribuem para o progresso científico.

Agradecimentos

Agradeço aos meus pais que sempre me apoiaram em minhas decisões e estiveram comigo ao longo da graduação, pelo sustento que me proporcionaram e pela paciência que tiveram.

Agradeço a meus amigos que fiz durante curso e que me proporcionaram diversas horas de momentos bons e engraçados, principalmente durante o último ano.

Agradeço a minha companheira, Ellen, que teve paciência e compreensão durante esse projeto e me proporcionou muitas conversas. Seu apoio e presença foram indispensáveis.

Por fim, agradeço a meus professores, orientadores e mentores de qualquer tipo que durante toda a graduação me proporcionaram diversas oportunidades e ensinamentos, me guiaram e forneceram o conhecimento que eu necessitava.

"O bom da ciência é que ela é verdadeira, quer você acredite nela ou não."

Neil deGrasse Tyson

Resumo

Atualmente, a *internet* possibilitou a conexão de pessoas ao redor do mundo e se tornou o principal meio de consumo e produção de informações. Porém, devido a sua velocidade de produção e grande quantidade de informações, muitos artigos e, principalmente, publicações em redes sociais contendo desinformações são propagados com rapidez. Com isso, diversos modelos de Inteligência Artificial para classificação de notícias falsas e desinformação foram estudados e elaborados, trazendo diversos resultados promissores. Entretanto, ainda faltam estudos que expliquem os motivos da classificação, ou seja, a explicabilidade desses modelos. A grande maioria de modelos estruturados como redes neurais são obscuros, ou seja, não é possível determinar os motivos que levaram à tomada de decisões. Este projeto buscou utilizar dois modelos de Aprendizado de Máquina com arquiteturas simples para essa tarefa de classificação com o intuito de explicar suas decisões. Para isso, foram utilizadas duas abordagens de explicabilidade: uma inerente do modelo de classificação e uma *post-hoc*. Foram testados dois conjuntos de dados, LIAR e FNID, que consistem de declarações relacionadas a figuras e acontecimentos políticos rotuladas pela sua veracidade. Por fim, é visualizado utilizando exemplos como certos vieses podem afetar as decisões de modelos mais simples.

Palavras-chave: Processamento de Linguagem Natural, Aprendizado de Máquina, Inteligência Artificial Explicável, Detecção de Desinformação, Detecção de Fake News.

Abstract

Today, the internet has made it possible to connect people around the world and has become the main means of consumption and information production. However, due to its speed of production and great amount of information, many articles and mainly publications in social networks containing misinformation are propagated quickly. With this, several models of Artificial Intelligence for classification of fake news and misinformation were studied and elaborated, bringing several promising results. However, there is still a lack of studies explaining the reasons for the classification, i.e., the explainability of these models. The vast majority of models structured as neural networks are obscure, that is, it is not possible to determine the reasons that led to the decision making. This project sought to use two Machine Learning models with simple architectures for this classification task in order to explain their decisions. For this, two approaches of explanability were used: one inherent of a classification model and one post-hoc. Two datasets were tested, LIAR and FNID, which consist of statements related to figures and political events labeled for their veracity. Finally, it is shown using examples how certain biases can affect the decisions of simpler models.

Keywords: Natural Language Processing, Machine Learning, Explainable Artificial Inteligence, Misinformation Detection, Fake News Detection.

Lista de figuras

Lista de quadros

Lista de tabelas

Lista de abreviaturas e siglas

AI	*Artificial Inteligence*
API	*Application Programming Interface*
CNN	*Convolutional Neural Network*
DL	*Deep Learning*
JSON	*JavaScript Object Notation*
LRP	*Layer-wise Relevance Propagation*
LSTM	*Long short-term memory*
ML	*Machine Learning*
NLP	*Natural Language Processing*
SS3	*Sequential S3 - Smoothness, Significance, Sanction*
XAI	*eXplainable Artificial Inteligence*

Sumário

1 Introdução

Desde o início da década de 2010 o campo da Inteligência Artificial (em inglês, *Artificial Intelligence*, AI) teve grande avanços na área de Aprendizagem Profunda (em inglês, *Deep Learning*, DL), especialmente em tarefas de reconhecimento de imagens e objetos (SCHMIDHUBER, 2015). Tais avanços foram possíveis graças ao aumento de dados e *hardwares* acessíveis em conjunto da popularização de *frameworks* e bibliotecas especializadas, permitindo com que pesquisadores e desenvolvedores entreguem modelos com maior rapidez (DOŠILOVIĆ; BRČIĆ; HLUPIĆ, 2018).

No período descrito acima, técnicas de Aprendizado de Máquina (em inglês, *Machine Learning*, ML) e AI foram aplicados no campo de Processamento de Linguagem Natural, (em inglês, *Natural Language Processing*, NLP) permitindo grandes avanços no estado da arte em criação de modelos de linguagem, análises morfológica e sintática, dentre outras tarefas. (OTTER; MEDINA; KALITA, 2018).

Apesar do progresso em diversas áreas e tarefas, devido à estrutura aninhada não linear das arquiteturas das redes, esses modelos são aplicados com uma abordagem de caixa-preta, ou seja, sem informações sobre o que gerou as predições e tomadas de decisões. Isso pode ser uma inconveniência em aplicações governamentais ou médicas, onde é necessária uma transparência na visualização ou interpretação dos resultados. Dessa forma, o campo de Inteligência Artificial Explicável (em inglês, *eXplainable Artificial Intelligence*, XAI) busca técnicas que melhor entendam o que os modelos desenvolvidos aprenderam, assim como técnicas que expliquem predições individuais (SAMEK; WIEGAND; MÜLLER, 2017).

Recentemente, com a popularização da comunicação através da *internet*, é possível identificar uma grande disseminação de desinformação (sendo também comumente usado o termo *fake news*), isto é, notícias falsas ou arquitetadas que atingem um alto grau de repercussão através de meios de comunicação e redes sociais. Essas notícias tem o potencial de afetar a opinião pública, assim como a possibilidade de alterar o resultado de eventos importantes, tais como decisões empresariais ou eleições (LAZER et al., 2018). Serviços de mensagens instantâneas e redes sociais potencializaram imensamente o alcance desses tipos de notícias. Graças a forma apelativa delas, sua dispersão é rápida (MONTEIRO et al., 2018).

Utilizando modelos de Aprendizado de Máquina para tarefas de NLP é possível abordar a tarefa de classificação de desinformação (LIU; WU, 2018; PÉREZ-ROSAS et al., 2017; RUCHANSKY; SEO; LIU, 2017) no formato de notícias ou postagens em redes sociais, apesar da tarefa não ser trivial (SHU et al., 2019) e ainda apresentar diversos desafios (MOHSENI; RAGAN; HU, 2019). O impacto social inerente da dispersão dessas notícias torna a tarefa relevante na atualidade.

Considerando a grande quantidade de dados gerados por usuários em redes sociais para a disseminação e consumo de notícias gera alto interesse, ainda mais considerando evidência do uso de contas falsas usadas para dispersão de notícias enviesadas durante as eleições nos Estados Unidos da América em 2016 (BESSI; FERRARA, 2016; ALLCOTT; GENTZKOW, 2017).

Logo, este trabalho busca avaliar duas técnicas de interpretabilidade na tarefa de classificação de desinformação, uma considerando um modelo de classificação de texto já arquitetado para explicabilidade e uma técnica de explicação de modelos estruturados iguais as redes neurais.

O Capítulo 2 apresenta as fundamentações teóricas necessárias para o projeto. Os Capítulos 2 e 3 contém a metodologia e desenvolvimento do projeto. Por fim, os Capítulos 4 e 5 mostram os resultados e conclusão obtidos, respectivamente.

2 Fundamentação teórica

2.1 Inteligência Artificial Explicável

> A habilidade de explicar o raciocínio atrás da decisão de um indivíduo para outras pessoas é um aspecto importante da inteligência humana. Não é só importante no contexto social (...), como também no contexto educacional, onde estudantes buscam entender o raciocínio de seus professores. (SAMEK; WIEGAND; MÜLLER, 2017, p.2, tradução nossa)

Gilpin et al. (2018) apontam que a definição do que é uma explicação expõe um debate filosófico sobre sua constituição. O foco é definir o que é necessário saber quando é perguntado um "por que?" a um algoritmo. Explicar algo depende do tipo de pergunta que é feita, de modo que, quando é preciso saber algo sobre um modelo é possível entender quantitativamente quando é obtida uma resposta, ou seja, quando não for mais possível continuar perguntando. Os autores ainda apontam que a explicação pode ser avaliada quanto a sua interpretabilidade e sua completude. O primeiro busca entender o funcionamento do modelo de modo que o conteúdo seja entendível a humanos. O segundo busca descrever o funcionamento de forma acurada, sendo completa a descrição de todos os parâmetros e equações naquele modelo.

Em geral, Inteligência Artificial Explicável pode ser dividida nas categorias de explicabilidade inerente e *post-hoc*. A primeira é obtida a partir da construção de modelos autoexplicatórios que podem incorporar o aspecto explicativo na sua construção, enquanto o segundo necessita de um segundo modelo criado para prover a explicação para o modelo existente (SHU et al., 2019).

Para que sistemas que utilizem modelos de ML sejam utilizados de forma massivas em aplicações do mundo real, satisfazer critérios como segurança, indiscriminação e direito a explicação são essenciais, dentre outros (DOSHI-VELEZ; KIM, 2018). Apesar disso, os autores notam que não é possível entender todos os modos de funcionamento de um sistema, assim então, caindo na questão do critério da interpretabilidade, explicar a razão das tomadas de decisão em relação a critérios estabelecidos. É ressaltado que não existe um consenso sobre a definição de interpretabilidade no campo de ML, caindo tipicamente entre explicações que são entendíveis de forma simples a humanos (sendo formas simplificadas ou aplicações) e alguma avaliação quantificada de uma representação. Ainda mais, é levantado a questão da generalização: um modelo de aprendizado de máquina treinado com certos dados e interpretado, será que ele vai generalizar para outra aplicação ou conjunto de dados?

Os autores acima listam uma série de fatores que podem estar relacionados para a generalização de uma explicação, divididos em duas categorias: fatores baseados em tarefas e fatores relacionados à explicação. A primeira série de fatores é com base na similaridade e

Quadro 1 – Motivações de indivíduos de interesses chaves na explicabilidade.

Indivíduo de Interesse	**Motivação**
Cientista de dados	Entender o modelo; retirar *bugs*; melhorar a performance.
Empresário	Entender o modelo; avaliar validade para o propósito; aceitar o uso.
Analista de risco de modelo	Desafiar o modelo; assegurar da robustez; aprovar o uso.
Regulador	Verificar impacto nos consumidores; verificar a confiabilidade.
Consumidor	Entender o impacto do modelo em sua vida e tomada de ações.

Fonte: Belle e Papantonis (2020)

necessidade da explicação: uma explicação global (do modelo como um todo) ou local, como é caracterizada a incompletude (qual o motivo que resulta na necessidade da explicação, seja esse nos dados de entrada ou no próprio entendimento de como um modelo funciona), a restrição de tempo e o quão o experiente o usuário é para entender o que foi interpretado do modelo. A segunda série de fatores é em relação a unidades básicas de explicação: que características formam essas unidades, quantas delas são necessárias, sua estruturação, interação entre unidades e o quão bem humanos podem entende-las.

Belle e Papantonis (2020) salientam que desenvolvedores de modelos de ML devem se preocupar com características de exatidão, robusteza (resistência a pequenas perturbações), enviesamento, possibilidade de melhoria, transferibilidade e compreensibilidade humana. O Quadro 1 mostra motivações de alguns indivíduos de interesse para obtenção de explicações, no cenário atual. Os autores também oferecem diferentes perspectivas sobre explicabilidade, categorizando por transparência, critério de avaliação e tipo de explicação. A transparência, a compreensão do funcionamento interno por um humano, é dividida em três aspectos:

a) Capacidade de simulação: capacidade de um ser humano reproduzir o funcionamento daquele modelo. Acaba sendo mais complicado em modelos complexos, como a maioria das redes neurais, levando somente modelos mais simples a pertencer a essa categoria;

b) Capacidade de decomposição: habilidade do modelo ser dividido em partes e essas poderem ser explicadas;

c) Transparência algorítmica: Entender o procedimento que leva um modelo tomar uma decisão ou gerar uma saída. De forma geral, modelos que conseguem ser analisados matematicamente entram nessa categoria.

Quanto ao critério de avaliação, os autores consideram a compreensibilidade, fidelidade, acurácia, escalabilidade e generalização como dimensões para se considerar em termos de explicabilidade, servindo como guias intuitivos para futuros avanços na área já que são difíceis de quantificar rigorosamente. Por fim, os tipos de explicações podem ser visuais, textuais, locais, por exemplo (representações dos dados de treinamento), por simplificação ou por relevância

Quadro 2 – Vantagens e desvantagens dos tipos de explicações.

Tipo de Explicação	Vantagens	Desvantagens
Explicações locais	Explica o comportamento do modelo em uma área local de interesse	Explicações não generalizam em uma escala global. Perturbações pequenas podem resultar em explicações diferentes. A definição de localidade é complexa. Algumas abordagens são instáveis.
Exemplificações	Exemplos representativos que proveem intuições sobre o funcionamento interno do modelo. Alguns algoritmos revelam os dados de treinamento que levam o modelo a suas predições.	Exemplos precisam de inspeção humana. Não ressaltam quais partes do exemplo influenciam o modelo.
Relevância das Características	Operam em cada instância do modelo, verificando a importância de cada característica de entrada na decisão do modelo. Várias abordagens propostas são acompanhadas de garantias teóricas.	São sensíveis em casos que as características são correlatas. Em vários casos, as soluções exatas são aproximadas, levando a efeitos indesejados, podendo afetar o resultado.
Simplificações	Modelos substitutos que explicam os incertos. Explicações resultantes, como regras, são facilmente entendíveis.	Modelos substitutos podem não estimar os modelos originais apropriadamente, além de terem suas próprias limitações.
Visualizações	De fácil comunicação com leigos. A maioria das abordagens são intuitivas e descomplicadas de implementar.	Existe um limite de quantas características podem ser representadas de uma vez. Humanos precisam inspecionar gráficos resultantes para produzir as explicações.

Fonte: Belle e Papantonis (2020)

das características dos dados de entrada (um método desse tipo é detalhado na Seção 2.4). O Quadro 2 mostra vantagens e desvantagens dos diferentes tipos de explicações avaliadas.

2.2 Classificação de Desinformação

Apesar de vários trabalhos, publicações e veículos de notícias utilizarem o termo *fake news* para indicar notícias falsas, diversos jornalistas optaram pelo desuso devido a sua popularização como ferramenta de desconfiança a informação por indivíduos (GENDREAU, 2017; MCMILLAN; WOOTSON JR., 2018) e até mesmo como "palavra da moda" (EGELHOFER et al., 2020). Para evitar esse problema e generalizar o tipo de informação lidado, fora das citações foi optado pelo uso do termo desinformação.

Allcott e Gentzkow (2017) definem *fake news* como artigos que são escritos intencionalmente falsos e que possam ser verificados, de forma que possam enganar seus leitores, tendo um foco especial nas eleições dos Estados Unidos da América em 2016. Causas para o aumento dessas notícias são discutidas, como a facilidade para produção de conteúdo relacionado em redes sociais e facilitada porta de entrada na industria de mídia. A simplicidade de monetização na internet através de plataformas de anúncio também atrai esse comportamento. Por fim, ressaltam que um aumento no índice de desconfiança nos meios de comunicação e crescente polarização política possam ter contribuído para a crescente produção desses tipos de artigos.

Vários trabalhos focam no aspecto de definição, métodos de obtenção e criação de conjuntos de dados e métricas para reconhecimento de notícias falsas para pesquisa, mas ignoram os efeitos que *feeds* e recomendadores de notícias tem na propagação, além da falta de categorização de fontes não confiáveis. É possível abordar o problema dividindo os métodos de detecção na criação de notícias (analise da composição), distribuição e consumo (atribuição

de fontes, padrões de propagação) (MOHSENI; RAGAN; HU, 2019). É também importante ressaltar que os dados de notícias muitas vezes são multimodais (podem incluir texto, vídeo, áudio, imagens e metadados, dentre outros), também dificultando a tarefa.

A classificação de desinformação é uma tarefa que procura rotular informações com base na sua veracidade (não necessariamente somente como falso e verdadeiro). Técnicas de Processamento de Linguagem Natural usadas para classificação desses tipos de notícias podem envolver o uso de redes convolucionais (LIU; WU, 2018), LSTMs (em inglês, *Long short-term memory*) (RUCHANSKY; SEO; LIU, 2017), *Naive Bayes* (GRANIK; MESYURA, 2017) e modelagem de características (PÉREZ-ROSAS et al., 2017).

Segundo Shu et al. (2019), nesse caso em redes sociais, a classificação de notícias falsas é uma tarefa desafiadora devido aos seguintes pontos:

a) *Fake news* são escritas com o intuito de enganar o leitor, sendo não trivial classificá-las baseada somente no contexto;

b) Dados provenientes de redes sociais são robustos e multi-modais, geralmente geradas por usuários e muitas vezes de forma anônima.

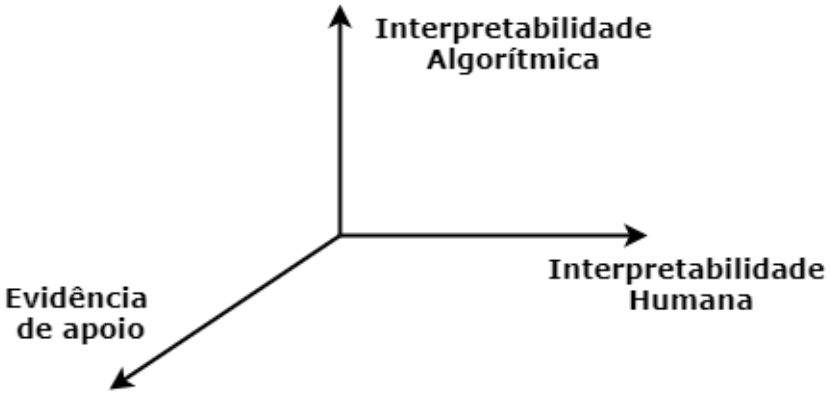
2.2.1 Classificação de Desinformação Explicável

Reis et al. (2019) mostram que muitos dos métodos de classificação de desinformação (no trabalho original, *fake news*) utilizam características de entrada relacionadas ao domínio da fonte (origem do artigo, metadados), sugerindo também que certos tipos de notícias falsas são identificadas por modelos com combinações específicas de características, mostrando a complexidade da tarefa de abordar todas os tipos de desinformação. Uma relação notada é que combinações de modelos podem ser um bom caminho de pesquisa.

Abordagens que utilizam diferentes perspectivas e combinações de modelos conseguem performar melhor que outras provendo explicações através do grau de importância de frases e palavras, assim como características linguísticas. Adicionalmente, representar visualmente as características aumentam a explicabilidade e facilitam a tomada de decisões (YANG et al., 2019).

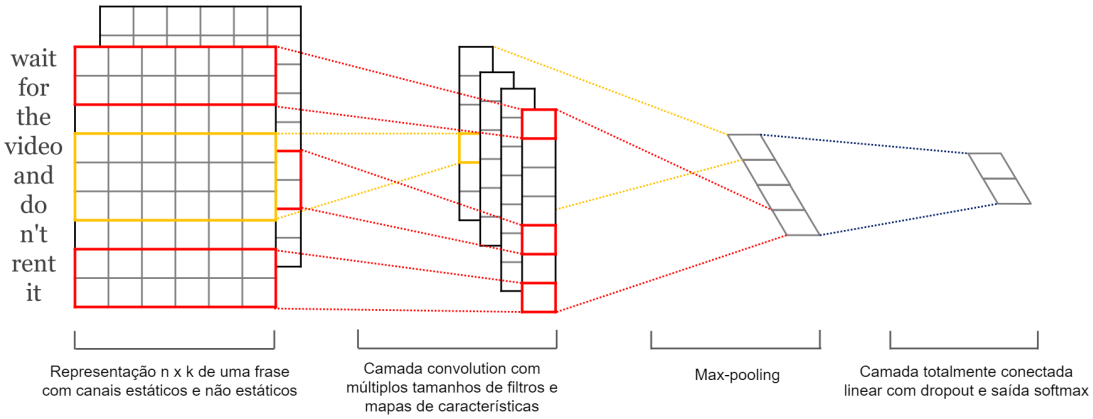
Mohseni, Ragan e Hu (2019) definem três dimensões da interpretabilidade para os diferentes propósitos em sistemas de detecção e mitigação de *fake news*, conforme indicado pela Figura 1. A interpretabilidade algorítmica é o grau de habilidade de pesquisadores e especialistas no campo de IA de visualizar os parâmetros do modelo e explicar seu comportamento. Interpretabilidade humana promove um grau de transparência na tomada de decisões, com explicações compreensíveis, especialmente para o usuário final. Por fim, no contexto de checagem de fatos, evidências de apoio são informações precisas que tem relação com o assunto das notícias e podem verificar sua veracidade, caso sejam inclusas nos dados de treinamento.

Figura 1 – Dimensões da interpretabilidade em sistemas de detecção de *fake news*.



Fonte: baseado em Mohseni, Ragan e Hu (2019).

Figura 2 – Modelo de arquitetura de uma CNN com dois canais para uma frase exemplo.



Fonte: baseado em Kim (2014).

2.3 Redes Neurais Convolucionais

Convolutional Neural Networks (CNNs) são um tipo de rede neural que utiliza camadas de filtros de convolução que são aplicados em características locais e criadas para tarefas de visão computacional (LECUN et al., 1998). Similarmente, CNNs podem ser usadas para classificação de sentenças e categorização de tópicos, mesmo com uma arquitetura simples (KIM, 2014). Em muitos casos, é utilizada uma CNN com uma estrutura 1D (considerando a ordem das palavras em um documento) dos documentos de texto no treinamento, sendo que cada unidade em uma camada de convolução é representada por uma sequencia de palavras de um documento (JOHNSON; ZHANG, 2015). A Figura 2 exemplifica a arquitetura de uma rede com uma camada convolucional utilizando *max-pooling* e *dropout*.

Kim (2014) descreve o funcionamento de uma rede convolucional para texto considerando uma palavra de posição i em uma frase como um vetor $x_i \in \mathbb{R}^k$, com dimensão k. Uma

concatenação de palavras sequenciais em uma frase é dada por $x_{i:i+j}$. A convolução envolvendo um filtro $w \in \mathbb{R}^{hk}$ sendo h a janela de palavras para formar uma característica c_i é dada por:

$$c_i = f(w \cdot x_{i:i+h-1} + b) \tag{2.1}$$

A função f é uma não-linearidade e b é um termo de viés. A convolução é então aplicada em cada janela possível em uma frase para produzir um mapa de características $c = [c_1, c_2, \ldots, c_{n-h+1}]$.

2.4 *Layer-wise Relevance Propagation*

Layer-wise relevance propagation (LRP) é uma técnica de explicação aplicada a redes neurais (ou modelos estruturados de forma semelhante), utilizável em entradas como imagens, vídeos ou texto. A partir de uma propagação reversa de uma função de predição $f(x)$ em uma rede neural, utilizando de regras de propagação locais (MONTAVON et al., 2019). O conceito geral do LRP é derivado da ideia de decomposição por pixel de imagens, ou seja, entender como cada pixel de uma imagem contribui para uma predição feita por um classificador. Em outras palavras, para cada imagem, separadamente, verificar quais pixels contribuem para um resultado de classificação positivo ou negativo (BACH et al., 2015).

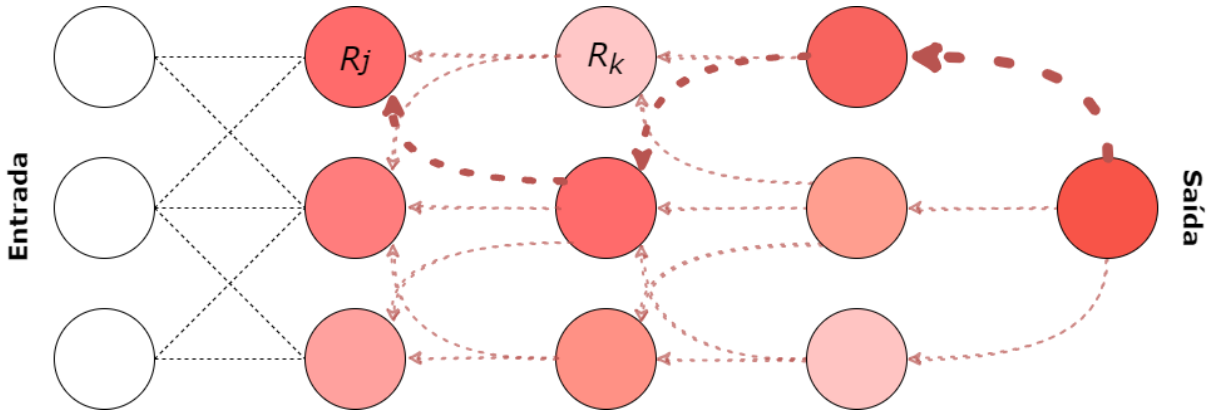
A ideia geral do LRP é distribuir, considerando cada classe possível em uma classificação, um valor de predição, um valor escalar, que causa a classificação, utilizando o processo de propagação no sentido inverso satisfazendo uma propriedade de conservação entre camadas. Cada camada intermediária até a camada de entrada recebe valores de relevância e a soma das relevâncias por camada é igual a pontuação de predição para a classe considerada (ARRAS et al., 2017a).

Segundo Montavon et al. (2019), o procedimento de propagação implementado pela técnica é sujeito a uma propriedade de conservação: o que foi recebido por um neurônio deve ser redistribuído para a camada anterior igualmente. Sendo j e k neurônios de duas camadas consecutivas em uma rede neural qualquer, a propagação de uma pontuação de relevância de um neurônio $(R_k)_k$ em uma camada anterior é feita aplicando a regra:

$$R_j = \sum_k \frac{z_{jk}}{\sum_j z_{jk}} R_k. \tag{2.2}$$

A quantidade z_{jk} é uma modelagem do quanto o neurônio j contribuiu para tornar o neurônio k relevante. A propriedade de conservação é demandada pelo denominador. A propagação termina quando a camada de entrada (as características de entrada) são atingidas. É possível verificar a propriedade de conservação local $\sum_j R_j = \sum_k R_k$ de cada camada e consequentemente a propriedade da conservação global $\sum_i R_i = f(x)$ (considerando a camada

Figura 3 – Exemplo do funcionamento do procedimento LRP.



Fonte: Elaborado pelo Autor.

de entrada i e o neurônio de saída contendo a predição). O funcionamento da rede é ilustrado na Figura 3: a quantidade recebida pelas camadas superiores por cada neurônio é distribuída a camada anterior. Cada neurônio contribui diferentemente em relação a força da suas ativações.

2.4.1 Regras de LRP

As regras a seguir foram apresentadas por Bach et al. (2015), exceto pela regra LRP-γ introduzida por Montavon et al. (2019), sendo compiladas no trabalho do mesmo. A notação considera o uso do LRP em redes retificadoras (funções de ativação ReLU), utilizada em uma gama de aplicações. Os neurônios dessas redes são compostos de:

$$a_k = max(0, \sum_{0,j} a_j w_{jk}), \tag{2.3}$$

onde o viés (*bias*) de cada neurônio é definido por w_{0k} e usando $a_0 = 1$. Três regras são apresentadas para essas redes.

Regra básica (LRP-0)

As contribuições são redistribuídas proporcionalmente de acordo com as contribuições de cada entrada para a ativação do neurônio conforme elas ocorrem na Equação 2.3, formulado como:

$$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k. \tag{2.4}$$

Coincidindo com conceitos de desativação, ausência de conexão e peso nulo, essa regra (assim como as regras abaixo) satisfaz propriedades básicas como $(a_j = 0) \vee (w_{j:} = 0) \Rightarrow R_j = 0$. Shrikumar, Greenside e Kundaje (2017) mostram que a aplicação dessa regra na rede

como um todo é equivalente ao produto do gradiente e da entrada. Gradientes de redes neurais costumam ser ruidosos, por isso a necessidade de outras regras para propagação.

Regra Épsilon (LRP-ϵ)

Uma melhoria da regra básica que adiciona um elemento positivo ϵ de tamanho pequeno no denominador:

$$R_j = \sum_k \frac{a_j w_{jk}}{\epsilon + \sum_{0,j} a_j w_{jk}} R_k. \tag{2.5}$$

Esse termo permite que relevâncias pequenas ou desprezíveis possam ser absorvidas por serem fracas ou contraditórias. Conforme valores maiores para ϵ, somente os valores de relevância mais salientes são repassadas. Essa regra permite representações de explicação que são mais esparsas nas características de entrada e menos ruidosas.

Regra Gama (LRP-γ)

A melhoria apresentada nessa regra é obtida pelo favorecimento de contribuições positivas aos neurônios sobre as contribuições negativas:

$$R_j = \sum_k \frac{a_j \cdot (w_{jk} + \gamma w_{jk}^+)}{\sum_{0,j} a_j \cdot (w_{jk} + \gamma w_{jk}^+)} R_k. \tag{2.6}$$

O parâmetro γ dita o quanto que as contribuições positivas são privilegiadas. Valores maiores de γ diminuem as contribuições negativas. A prevalência de contribuições limita o crescimento das relevâncias na propagação pela rede. Isso permite explicações que sejam mais estáveis. O tratamento das propagações positivas e negativas de forma assimétrica foi proposta por Bach et al. (2015) como a regra LRP-$\alpha\beta$:

$$R_j = \sum_k (\alpha \frac{(a_j w_{jk})^+}{\sum_{0,j} (a_j w_{jk})^+} - \beta \frac{(a_j w_{jk})^-}{\sum_{0,j} (a_j w_{jk})^-}) R_k. \tag{2.7}$$

Com os parâmetros α e β é possível controlar separadamente, respectivamente, o favorecimento das contribuições positivas e das contribuições negativas. Esses parâmetros estão sujeitos a restrição de conservação $\alpha = \beta + 1$.

2.4.2 Utilização das regras de LRP

A discussão sobre a interpretação de diversos modelos e sistemas de aprendizado de máquina é um tópico ainda muito discutido, como visto na Seção 2.1. Conforme visto na Subseção 2.4.1, o sistema estruturado pelo LRP permite bastante flexibilidade na escolha de regras. Para avaliação das vantagens e desvantagens de cada regra, Montavon et al. (2019)

verificaram para a tarefa de classificação de imagens que utilizar um conjunto de regras de LRP na rede, isto é, utilizar diferentes regras para diferentes camadas, é possível uma representação mais fidedigna e entendível. Ainda mais, esclarece:

> De forma geral, para aplicar LRP com sucesso em uma nova tarefa, é importante verificar cuidadosamente as propriedades das camadas da rede neural e verificar com o ser humano qual tipo de explicação é mais entendível para o tal. (MONTAVON et al., 2019, p.203, tradução nossa)

Em modelos de processamento de linguagem natural que utilizam de LRP para explicação, foi utilizado o LRP uniforme, ou seja, uma regra de LRP para a rede inteira, como LRP-0 (ARRAS et al., 2017a) e LRP-ϵ (POERNER; ROTH; SCHÜTZE, 2018; ARRAS et al., 2016; ARRAS et al., 2017b).

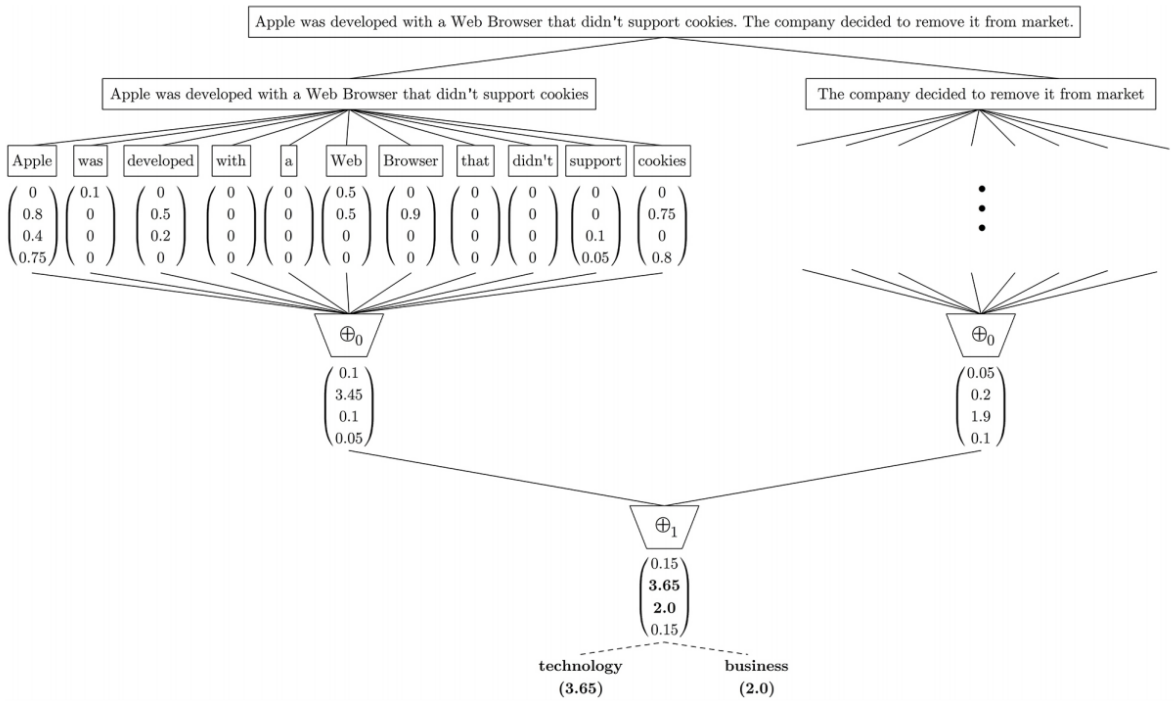
2.5 *Sequential S3*

O modelo de Aprendizado de Máquina *Sequential S3* (SS3), que significa "*Smoothness, Significance and Sanction*" (Suavidade, Significância e Sanção), é um classificador de texto com a capacidade de naturalmente explicar suas justificativas (BURDISSO; ERRECALDE; GÓMEZ, 2019b). A descrição do modelo e seu funcionamento a seguir foram apresentados pelos autores originais. O modelo foi proposto para tarefas de detecção de risco prévias, especificamente para detecção de depressão. O modelo foi projetado com o intuito de suportar aprendizado incremental, classificação prévia (isto é, a decisão de quando parar de avaliar o fluxo de entrada e classificar com uma acurácia aceitável) e explicabilidade. O método foi projetado para ser usado como um modelo estruturado geral para classificação de texto, sendo flexível o suficiente para diferentes tarefas, dependendo do problema.

O processo de classificação depende de uma função $gv(w, c)$ que avalia palavras em relação a categorias. Em outras palavras, a função gv recebe uma palavra w e uma categoria c e resulta em uma pontuação entre 0 e 1. Detalhes de como os valores são obtidos pela função estão na Subseção 2.5.1. Essa pontuação representa o grau de confiança com o qual a palavra w é possivelmente exclusiva a pertencer à categoria c. A expressão $gv(w, c) = v$ é lida como "w possui um valor global v na categoria c". Ainda é possível definir que $gv(w) = (gv(w, c_0), gv(w, c_1), \ldots, gv(w, c_k))$ onde $c_i \in C$, sendo que C é o conjunto de todas as categorias. Dessa forma, o vetor $gv(w) = \vec{v}$ é um vetor de confiança para a palavra w.

O processo de classificação pode ser dividido em duas partes. A primeira fase depende da divisão de cada documento em blocos (de forma prática, documentos são divididos em parágrafos, parágrafos em frases e frases em palavras). Ao final, existe uma hierarquia de blocos onde as palavras estão no nível mais baixo e os parágrafos estão no nível mais alto. Para cada palavra, a função gv é aplicada para obter os vetores de confiança. Esses vetores são então reduzidos por meio de um operador de sumário (por exemplo, soma, máximo, mínimo, ...).

Figura 4 – Processo de classificação do SS3 para um documento de texto.



Fonte: Burdisso, Errecalde e Gómez (2019b)

Esses vetores geram os vetores de confiança do nível acima. Cada nível da hierarquia pode utilizar um operador diferente. Por fim, a classificação é feita com base no único vetor de confiança gerado, com base em alguma estratégia (por exemplo, utilizar o valor do vetor com maior valor). O funcionamento da hierarquia de blocos pode ser vista na Figura 4, mostrando sua divisão e computação dos vetores de confiança. No final, as classes com maior probabilidade são '*technology*' e '*business*'.

A computação dos vetores de confiança também é incremental, desde que o operador de sumário suporte operações incrementais (o que é o caso para adição, multiplicação, máximo ou média). Com essa abordagem incremental é possível também observar como esse vetor muda com o tempo, podendo derivar regras simples e claras de quando um modelo deve fazer uma classificação prévia.

2.5.1 Definição formal

Essa seção apresenta as definições propostas por Burdisso, Errecalde e Gómez (2019b) para a fundação das intuições descritas na seção anterior. Adicionalmente, a definição da variação do modelo chamada τ-SS3 que expande o aprendizado do modelo utilizando n-gramas (BURDISSO; ERRECALDE; GÓMEZ, 2020) é também descrita. As definições utilizadas foram retratadas em ambos trabalhos.

Cálculo do valor local e global de palavras

A abordagem utilizada para calculo do gv de uma palavra tenta superar problemas que resultam da tentativa de calcular um valor de uma palavra baseado somente na sua informação local em uma categoria. Um valor local lv é calculado para cada categoria e só então eles são combinados entre todas as categorias para cálculo de gv. O valor local de uma palavra em uma categoria deve ser proporcional a probabilidade da de w em c. O valor local fica definido como:

$$lv(w,c) = \frac{P(w|c)}{P(w_{max}|c)}. \tag{2.8}$$

A vantagem da divisão pela probabilidade da palavra mais frequente, ao invés de utilizar somente $lv(w,c) = P(w|c)$, produz efeitos positivos como a normalização do valor local (com a palavra mais provável tendo valor 1) e palavras são avaliadas de acordo com sua proximidade à palavra mais provável. Ou seja, palavras vazias (*stopwords*, em inglês) terão valores próximos de 1. De forma geral, para não assumir que a proporcionalidade $lv(w,c) \propto P(w|c)$ diretamente, a definição para o valor local é:

$$lv_\sigma(w,c) = \left(\frac{tf_{w,c}}{max\{tf_c\}}\right)^\sigma, \tag{2.9}$$

onde $tf_{w,c}$ indica a frequência da palavra w na categoria c e $max\{tf_c\}$ a frequência máxima vista na categoria c. O valor de $\sigma \in (0,1]$ é o hiper-parâmetro de suavidade do modelo.

Esse primeiro hiper-parâmetro indica a relação entre a frequência natural e o valor final atribuído à palavra, ou seja, controla o crescimento do valor local da palavra em relação a quão perto está de uma palavra mais provável. Seu ajuste consegue remover o efeito de obstrução que palavras frequentes (mas não tão importantes, como as palavras vazias) produzem em palavras menos frequentes que são mais importantes.

Com a capacidade de calcular valores locais é possível definir o valor global:

$$gv(w,c) = lv_\sigma(w,c) \cdot sg_\lambda(w,c) \cdot sn_\rho(w,c). \tag{2.10}$$

As funções sg e sn utilizam dos hiper-parâmetros do modelo $\lambda, \rho \in \mathbb{R}^+$, respectivamente, referindo-se a significância e a sanção. Ambas são funções da forma $f : W \times C \mapsto [0,1]$.

$$sg_\lambda(w,c) = \frac{1}{2}\, tanh\left(4\,\frac{(lv(w,c) - \widetilde{LV}_w)}{\lambda \cdot MAD_w} - 2\right) + \frac{1}{2}. \tag{2.11}$$

Temos que $LV_w = \{lv(w,c_i)|c_i \in C\}$, ou seja, o conjunto de todos os valores locais para a palavra w, sendo $\widetilde{LV}_w$ a mediana do conjunto. O desvio absoluto (utilizando a mediana como tendencia central) de LV_w é denotado por $MAD_w = median(|lv(w,c_i) - \widetilde{LV}_w|)$. O hiper-parâmetro λ controla o quão longe o valor local deve variar para ser considerado

significante. Dessa forma, esse fator captura a significância global da palavra, diminuindo seu valor global em relação ao seu valor local em outras categorias.

A função de sanção sn diminui proporcionalmente o valor global de uma palavra w em relação ao número de categorias para qual w é significante:

$$sn_\rho(w, c) = \Big(- \frac{\hat{C}_{wc}}{|C| - 1} + \Big)^\rho. \tag{2.12}$$

O número de categorias é denotado por $|C|$ e $\hat{C}_{wc} = \sum_{c_i \in C - \{c\}} sg_\lambda(w, c_i)$. Esse termo é a sumarização de todas as categorias em C fora c. Dessa forma, ρ, o último hiper-parâmetro do modelo, permite controlar a severidade da sanção em proporção ao número de categorias significantes.

τ-SS3

Uma fraqueza do modelo SS3 original é o não reconhecimento da relação entre palavras individuais. Dessa forma, o modelo pode acabar não percebendo algumas sequências que poderiam melhorar os resultados de classificação. A extensão τ-SS3 (BURDISSO; ERRECALDE; GÓMEZ, 2020) permite a identificação de n-gramas para classificação de documentos.

A única mudança necessária na definição formal é uma generalização da Equação 2.8, permitindo também a obtenção de valores locais a partir de sequências de k palavras, isto é, $t_k = w_1 \rightarrow w_2 \ldots \rightarrow w_k$, definindo lv como:

$$lv_\sigma(t_k, c) = \Big(\frac{P(w_1 w_2 \ldots w_k | c)}{P(m_1 m_2 \ldots m_k | c)} \Big)^\sigma. \tag{2.13}$$

O denominador indicado por $m_1 m_2 \ldots m_k$ é a sequência de k palavras com a maior probabilidade de ocorrer. A definição utilizada para o calculo de lv é então:

$$lv_\sigma(t_k, c) = \Big(\frac{tf_{t_k,c}}{max\{tf_{k,c}\}} \Big)^\sigma, \tag{2.14}$$

onde $tf_{t_k,c}$ denota a frequência da sequência t_k em c e $max\{tf_{k,c}\}$ a frequência máxima na categoria c para uma sequência de tamanho k.

3 Metodologia

Este capítulo descreve a estrutura de desenvolvimento de um projeto de Aprendizado de Máquina e apresenta as ferramentas utilizadas.

3.1 Etapas

O progresso desse trabalho consistiu das seguintes etapas: levantamento de estudos sobre explicabilidade, Inteligência Artificial Explicável, classificação de desinformação e métodos relacionados. Junto com o levantamento de estudos, foram pesquisados conjuntos de dados de notícias rotuladas para a tarefa de classificação.

Dois modelos foram escolhidos para classificação, sendo um modelo arquitetado para explicabilidade (SS3) e outro sendo uma abordagem *post-hoc* (LRP) aplicada a uma rede neural simples. Por fim, foram colhidos os resultados do treinamento dos modelos e as técnicas de explicabilidade foram avaliadas a partir da perspectiva da interpretabilidade humana. As etapas são descritas em mais detalhes no Capítulo 4.

3.2 Ferramentas

3.2.1 NumPy

NumPy (HARRIS et al., 2020) é um pacote Python *open-source* para computação científica que oferece um objeto *array* multidimensional, além de rotinas implementadas para execução rápida nessas estruturas como operações lógicas, matemáticas, ordenação, álgebra linear, dentre outras. A capacidade de *broadcasting* permite realizar operações entre *arrays* de tamanhos diferentes (dado que o menor deles seja expansível), descrevendo implícitamente o comportamento elemento por elemento.

3.2.2 PySS3

PySS3 (BURDISSO; ERRECALDE; GÓMEZ, 2019a) é um pacote *open-source* construído em Python que implementa o método SS3 (seu funcionamento é descrito na Seção 2.5) incluindo ferramentas de visualização e interatividade. Essas ferramentas permitem a análise, monitoramento e compreensão do funcionamento dos modelos. Seu funcionamento e API (em inglês, *Application Programming Interface*) são similares a modelos do scikit-learn. O pacote também implementa a variação τ-SS3.

A classe `Live_Test` permite o teste do modelo e visualização do processo de classificação que o modelo SS3 realiza utilizando uma ferramenta interativa no navegador que permite também o teste de novos documentos. A Classe `Evaluation` oferece métodos de avaliação (como *k-fold*) utilizando várias métricas (acurácia, precisão, *recall*, dentre outras), além de oferecer um gráfico 3D interativo representando diversos experimentos conforme a variação de hiper-parâmetros. O modelo ainda inclui um quarto parâmetro conhecido como α, que permite definir o valor global mínimo que uma palavra deve ter para a classificação nela ocorra, ou seja, somente palavras onde $gv > \alpha$ serão consideradas durante a classificação.

3.2.3 iNNvestigate

iNNvestigate (ALBER et al., 2019) é um pacote *open-source* que oferece uma interface comum para vários métodos de explicabilidade *post-hoc* (como o *layer-wise relevance propagation*), sendo aplicado a um grande número de classes de redes neurais, sendo simples de inicializar e qualitativamente comparar métodos. Sua implementação toma proveito de cálculos já feitos por outras bibliotecas nas questões das propagações para frente e para trás em uma rede neural, evitando a redundância de código.

3.2.4 Keras

Keras (CHOLLET et al., 2015) é uma biblioteca *open-source* que oferece uma interface Python para redes neurais, também atuando como interface para a biblioteca TensorFlow. Diversos blocos de modelos comuns de redes neurais são implementados, assim como funções de ativação e otimizadores, assim como ferramentas de utilidade para tarefas que lidam com imagem e texto, tornando mais fácil e rápida a implementação de modelos de aprendizado de máquina e inteligência artificial.

3.2.5 pandas

pandas (MCKINNEY, 2010) é uma biblioteca *open-source* de software para a linguagem Python que implementa estruturas de dados flexíveis e rápidas que lidam com dados relacionais ou rotulados, de forma fácil e intuitiva. Sua manipulação, em particular, permite a manipulação de tabelas numéricas e séries temporais. Seu principal objeto para manipulação de dados é o `DataFrame`, permitindo indexação. Com ele, diversas operações podem ser realizadas como reformatação, leitura e escrita em arquivos alinhamento de dados, divisão de colunas e linhas, dentre outras.

3.2.6 scikit-learn

Scikit-learn (PEDREGOSA et al., 2011) é uma biblioteca *open-source* para a linguagem Python. Ela oferece algorítimos de regressão linear, classificação e agrupamento como SVM

Quadro 3 – Matriz de confusão.

Previsto\Real	**Positivo**	**Negativo**
Positivo	TP	FP
Negativo	FN	TN

Fonte: Elaborado pelo autor.

(em inglês, *Support Vector Machine*), florestas aleatórias, *k-means*, dentre outros. Esse pacote foi feito para operar juntamente com as bibliotecas NumPy e SciPy.

3.2.7 Jupyter Notebook

Jupyter Notebooks (KLUYVER et al., 2016) são ambientes interativos para web suportando várias linguagens permitindo a criação de documentos, implementado como um documento JSON (*JavaScript Object Notation*), com suporte a versionamento, contendo uma lista de células que podem conter código texto, textos matemáticos, imagens, dentre outros. Graças a natureza interpretável da linguagem, os códigos podem ser editados e executados dependo da ordem das células.

3.2.8 Matplotlib

Matplotlib (HUNTER, 2007) é uma biblioteca implementada em Python de formulação e desenho 2D e 3D de imagens (especialmente para publicações científicas) de qualidade em vários tipos de formas e formatos, podendo ser usado em código Python assim como em Jupyter Notebooks. É possível gerar gráficos, histogramas e outros com poucas linhas de código.

3.3 Métricas

Para quantificação da eficácia nas predições são necessárias métricas, uma vez que após o treinamento é necessário uma avaliação do comportamento dos modelos. Para definir as mensurações é utilizada uma matriz de confusão. É uma matriz quadrada $2x2$ utilizada para comparação das classes reais com as classes preditas, seguindo a notação:

a) TP - *True Positive* ou Verdadeiro Positivo: Saída definida positiva corretamente;

b) TN - *True Negative* ou Verdadeiro Negativo: Saída definida negativa corretamente;

c) FP - *False Positive* ou Falso Positivo: Saída definida positiva incorretamente;

d) FN - *False Negativo* ou Falso Negativo: Saída definida negativa incorretamente.

O Quadro tal mostra como tais quantidades são representadas na matriz. Dessa forma, é possível definir as medidas que permitem algumas visões de desempenho dos modelos:

a) Acurácia $= \frac{TP+TN}{TP+TN+FP+FN}$;

b) Precisão $P = \frac{TP}{TP+FP}$;

c) Sensitividade $S = \frac{TP}{TP+FN}$;

d) Pontuação F1 $= \frac{2 \cdot P \cdot S}{P+S}$.

4 Desenvolvimento

Esse capítulo trata do detalhamento das etapas do desenvolvimento desse projeto.

4.1 Obtenção dos dados

Nesta etapa foram realizadas pesquisas de conjuntos de dados (*datasets*) de notícias (generalizando para desinformação) rotuladas baseadas na suas veracidades. Ambos conjuntos de dados são coletados de declarações do site PolitiFact[1], vencedor do premio Pulitzer em 2009. O site convém tópicos políticos, introduzindo informações, julgamentos e checagem de fatos sobre as declarações, incluindo fontes e rótulos diversos. Todos os conjuntos de dados foram elaborados e divididos em conjuntos de treinamento, validação e teste.

4.1.1 LIAR

LIAR (WANG, 2017) é um conjunto de dados desenvolvido para classificação de desinformação em inglês de declarações manualmente rotuladas obtidas de PolitiFact, que provem referências, análises e fontes para cada caso, de forma que a checagem de fatos possa ser feita. Em comparação com conjuntos de dados desenvolvidos com *crowdsourcing* (construídos com colaborações de terceiros) que quando testados em situações do mundo real não generalizam bem.

As declarações do conjunto são rotuladas seis categorias em uma escala de veracidade: *pants-fire*, *false*, *barely-true*, *half-true*, *mostly-true*, e *true*. A Tabela 1 mostra estatísticas de tamanho do conjunto e a Tabela 2 mostra a quantidade de dados por rótulo.

Tabela 1 – Tamanho dos conjunto de dados no *dataset* LIAR.

Conjunto de treinamento	10269
Conjunto de validação	1284
Conjunto de teste	1283
Tamanho médio de *token*	17.9

Fonte: baseado em Wang (2017).

4.1.2 FNID

FNID (*Fake News Inference Dataset*) (AMIRKHANI, 2020) é um conjunto de dados criado para a tarefa de Inferência de Linguagem Natural (tarefa de determinar se uma hipótese

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

Tabela 2 – Quantidade de rótulos para cada conjunto no *dataset* LIAR.

	Conjunto		
Rótulo	**Treinamento**	**Validação**	**Teste**
pants-fire	839	116	92
false	1995	263	249
barely-true	1654	237	212
half-true	2114	248	265
mostly-true	1962	251	241
true	1676	169	208

Fonte: Elaborado pelo autor.

é verdadeira, falsa ou indeterminada dada uma premissa). O conjunto foi criado também utilizando declarações obtidas do PoliticFact. Foi criado para ser compatível com os *datasets* LIAR e FakeNewsNet (FNN) (SHU et al., 2018) para avaliação e análise, composto de dois conjuntos, nomeados FNID-LIAR e FNID-FNN, respectivamente. FNID-LIAR é composto dos mesmos rótulos que LIAR e FNID-FNN é composto de somente dois, verdadeiro e falso. As Tabelas 3 e 4 mostram estatísticas dos conjuntos de dados dentro do *dataset* FNID.

Tabela 3 – Tamanho dos conjunto de dados no *dataset* FNID.

FNID-LIAR	
Treinamento	15052
Validação	1265
Teste	1266
FNID-FNN	
Treinamento	15212
Validação	1058
Teste	1054

Fonte: baseado em Amirkhani (2020).

Tabela 4 – Quantidade de amostras para cada conjunto no *dataset* FNID.

Rótulo	**Treinamento**	**Validação**	**Teste**
	FNID-LIAR		
pants-fire	1775	145	92
false	3280	280	249
barely-true	2483	202	212
half-true	2833	240	266
mostly-true	2631	226	239
true	2050	172	208
	FNID-FNN		
fake	7621	518	418
real	7591	540	636

Fonte: Elaborado pelo autor.

4.2 Preparação dos dados

Para os modelos que operam dentro do contexto do Processamento de Linguagem Natural é essencial o pré-processamento dos documentos de texto. Para interpretação dos textos é necessária a padronização das palavras para formação de *tokens* (pequenas unidades de texto) que então serão utilizados no aprendizado. O pacote NLTK[2] possui várias funções que auxiliam em tarefas de NLP, especialmente nessa etapa.

Dos modelos utilizados, apesar do funcionamento do pré-processamento ser diferente para cada um, seu processo é semelhante. Os procedimentos utilizados no trabalho foram:

a) Remoção de caracteres numéricos, espaços e acentuações[3];

b) Transformação de todos os caracteres em minúsculos;

c) Remoção de palavras vazias (*stopwords*), palavras frequentes nos documentos mas com pouca importância como *o*, *a*, *em*, e *no*;

d) *Stemming* das palavras, ou seja, redução em lexemas. Um lexema é a unidade mínima do distintiva do sistema semântico de uma língua que utiliza várias flexões para uma mesma palavra. Nesse passo foi utilizado o Porter Stemmer[4].

Para o conjuntos LIAR e FNID-LIAR, para padronização com a quantidade de rótulos do FNN, seus rótulos foram agrupados nas categorias *falso* e *verdadeiro*.

Foi feita uma análise das palavras mais frequentes em cada conjunto de dados. Devido a origem dos conjuntos de dados, as palavras com maior ocorrência são as mesmas para os três *datasets*. A frequência dos *tokens* pode ser vista na Figura 5. Para facilitar o manejar dos dados, foram criados dois *loaders* (objetos em código Python que implementam a leitura dos dados no disco, mais precisamente de arquivos `.tsv` e `.csv`) para os conjuntos LIAR e FNID, respectivamente.

4.3 Escolha dos modelos

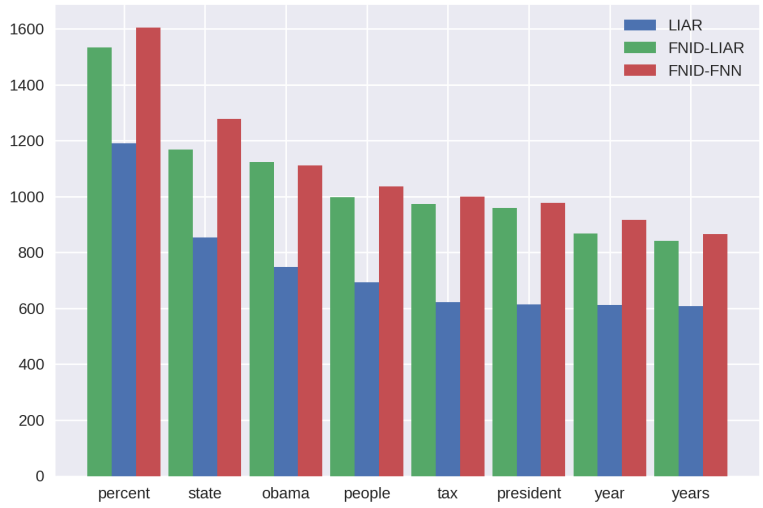
Este projeto utilizou dois modelos para a classificação utilizando os *datasets* citados:

a) SS3 (descrito na Seção 2.5);

b) Rede neural convolucional 1D. A arquitetura da rede convolucional pode ser vista na Figura 6.

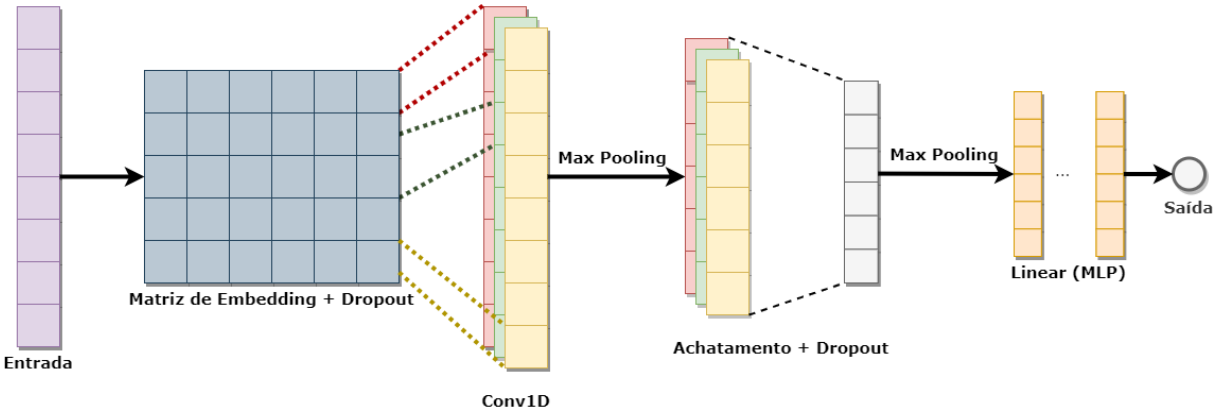
[2] https://www.nltk.org/.
[3] Foi utilizada a expressão regular `^a-zA-Z` para seleção.
[4] https://tartarus.org/martin/PorterStemmer/. Também disponível no NLTK

Figura 5 – Frequência dos *tokens* em cada conjunto de dados utilizado.



Fonte: Elaborado pelo autor.

Figura 6 – Arquitetura da rede convolucional usada.



Fonte: Elaborado pelo autor.

Figura 7 – Acurácia no conjunto de validação nos *datasets* avaliados.

Fonte: Elaborado pelo autor.

4.4 Treinamento dos modelos

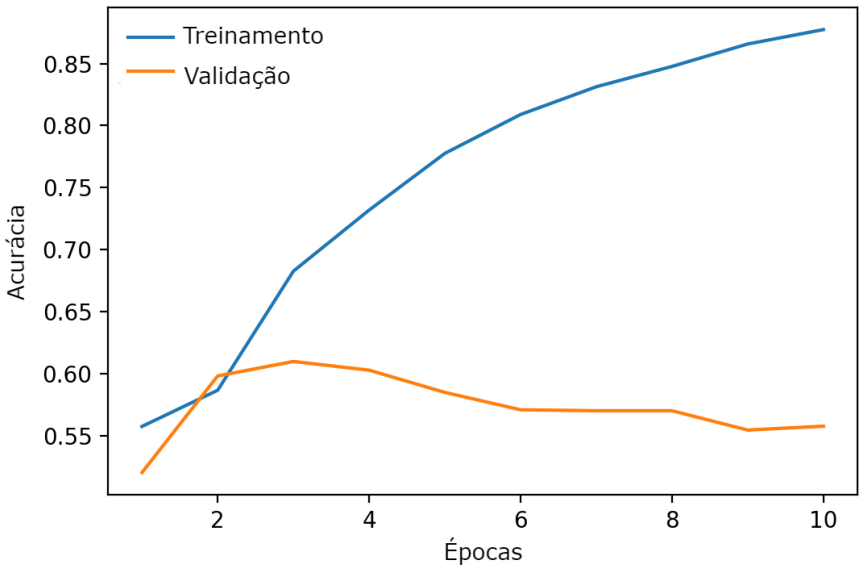
Para primeiramente avaliar a operação dos modelos, foi realizado o treinamento utilizando cada conjunto de treino e então realizada uma avaliação utilizando os conjuntos de validação de cada *dataset*.

No modelo S33, foram realizados testes utilizando o modelo padrão e utilizando 2-gramas e 3-gramas (*n-grams*), entretanto não foram obtidas diferenças na acurácia em nenhum dos conjuntos de validação, dessa forma, todos os resultados são do modelo sem utilização de aprendizado de sequências. Nessa etapa, os hiper-parâmetros ainda não foram otimizados. As acurácias de validação para cada modelo estão disponíveis na Figura 7.

Na rede convolucional, a camada de convolução possui ativação ReLu e inicialmente foram utilizados 8 filtros em uma janela de largura 3. Os modelos foram treinados por 10 épocas. Entretanto, conforme visto pela Figura 8, a acurácia no conjunto de treinamento foi aumentando, enquanto a mesma diminuía no conjunto de validação, indicando *overfitting*, ou seja, o modelo acaba se ajustando muito bem para os dados de treinamento, mas não generaliza para novos dados. Dessa forma, para os parâmetros até o momento, o conjunto deveria ser treinado até as épocas 3 e 4, para os conjuntos LIAR e ambos contemplados pelo FNID, respectivamente. Depois da otimização, o número de épocas de treinamento foi mantido o mesmo para evitar *overfitting*.

Nos modelos convolucionais, os rótulos são binários com 0 representando uma notícia verdadeira e 1 uma notícia falsa.

Figura 8 – Comparação da variação da acurácia de treinamento e validação no conjunto LIAR indicando *overfitting*.



Fonte: Elaborado pelo autor.

4.4.1 Otimização dos hiper-parâmetros

Utilizando as análises do conjunto de validação foram otimizados os hiper-parâmetros dos modelos.

No modelo SS3, foi realizada uma busca em grade (*grid-search*) dos hiper-parâmetros que maximizavam a acurácia no conjunto, com a seguinte variação para todos os conjuntos:

a) σ: Intervalo de variação entre 0 e 1 com passo 0.03125;

b) λ: Intervalo de variação entre 0 e 2 com passo 0.06250;

c) α: Intervalo entre 0 e 0.1 com passo 0.01.

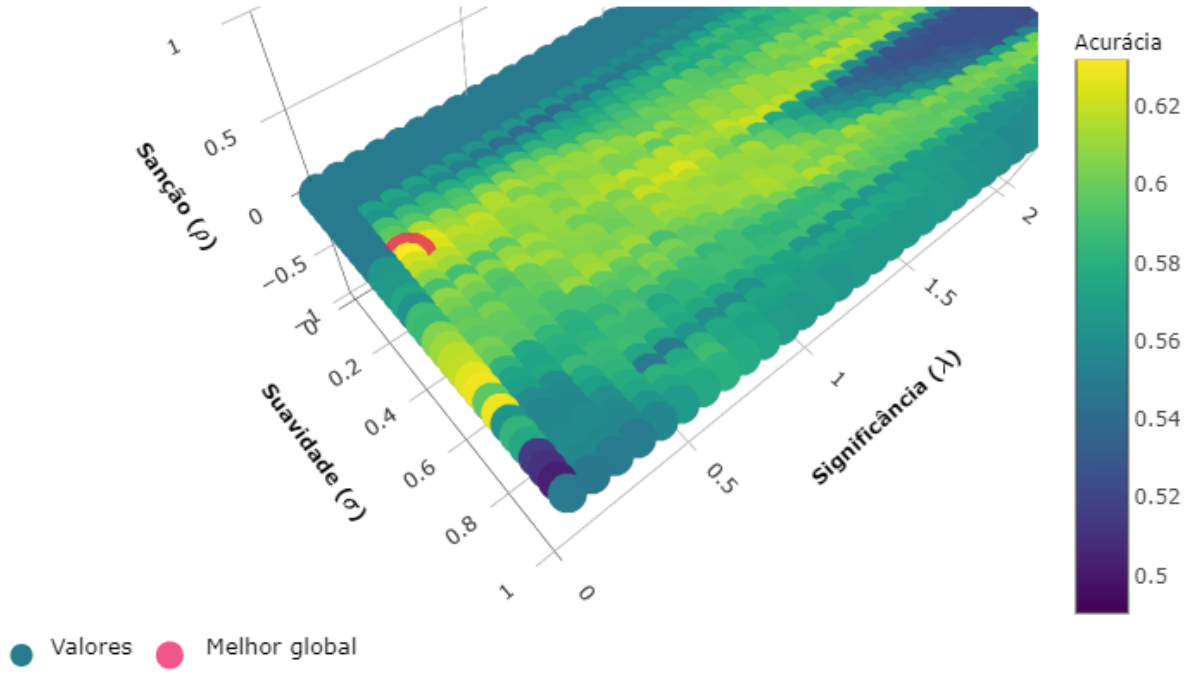
A Figura 9 mostra todos os testes feitos na busca. O parâmetro ρ foi mantido com valor 0 devido ao problema de classificação ser binário e sua variação não alterar o comportamento do modelo. A Tabela 5 mostra os hiper-parâmetros que resultaram em maior acurácia nesse modelo.

Tabela 5 – Hiper-parâmetros do modelo SS3 escolhidos para cada conjunto.

	LIAR	**FNID-LIAR**	**FNID-FNN**
σ	0.6667	0.5333	0.3226
λ	1.447	1.4348	0.0645
α	0.0222	0	0

Fonte: Elaborado pelo autor.

Figura 9 – Busca em grade no conjunto de validação do *dataset* FNID-FNN para otimização de hiper-parâmetros.



Fonte: Elaborado pelo autor.

O modelo CNN também teve seus hiper-parâmetros otimizados, sendo os mesmos para todos os conjuntos de dados. As camadas de regularização com *dropout* tem ativação com frequência de 0.3. A camada de *embedding* é representada por uma matriz de altura 40 e larguras 20 e 25 para os conjuntos LIAR e FNID, respectivamente. A camada de convolução utiliza então 16 filtros com uma largura de 3 *tokens*. Para atualizar a rede é utilizado o otimizador Adam e a função de custo é a entropia cruzada binária.

4.5 Resultados e avaliação dos modelos

Nessa etapa foi utilizado o conjunto de teste para análise dos modelos. Os modelos foram treinados com cada conjunto de treinamento com os parâmetros encontrados na otimização e avaliados com o conjunto de teste.

As Tabelas 6 e 7 mostram os resultados obtidos para cada modelo em todos os *datasets*. Para a CNN, os valores são médias calculadas com vários testes. As matrizes de confusão dos modelos em todos os conjuntos são representadas pelas Figuras 10 e 11.

É possível observar que, para modelos simples, sua performance é razoável, mas deixa a desejar comparado a arquiteturas de redes neurais mais complexas que utilizam mais dados

Tabela 6 – Resultados dos modelos SS3 em relação ao conjunto de teste

	Acurácia	**Precisão**	**Sensitividade**	**Pontuação F1**
LIAR	0.6133	0.6035	0.5953	0.5941
FNID-LIAR	0.5987	0.5976	0.5981	0.5900
FNID-FNN	0.6395	0.7219	0.6910	0.6354

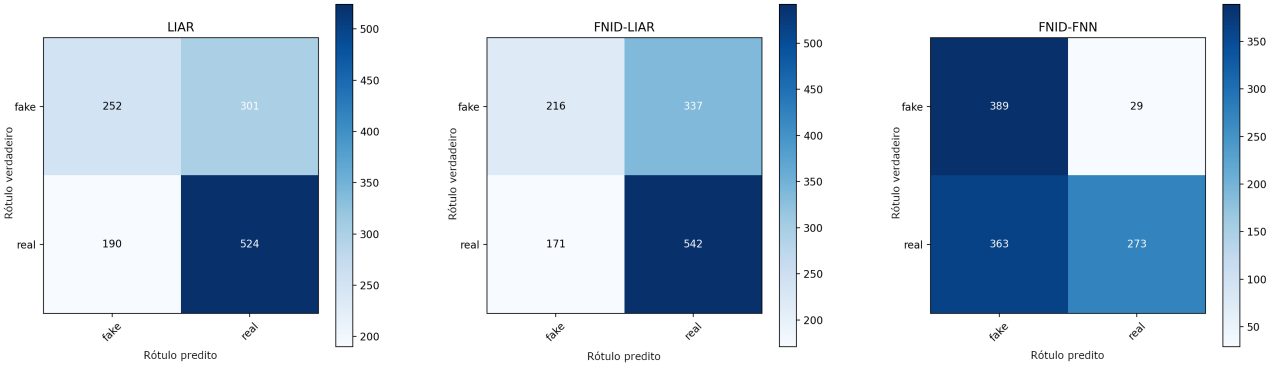
Fonte: Elaborado pelo autor.

Tabela 7 – Resultados dos modelos CNN em relação ao conjunto de teste

	Acurácia	**Precisão**	**Sensitividade**	**Pontuação F1**
LIAR	0.6196	0.5877	0.4363	0.4919
FNID-LIAR	0.5940	0.5394	0.5381	0.5295
FNID-FNN	0.6850	0.5134	0.6488	0.5203

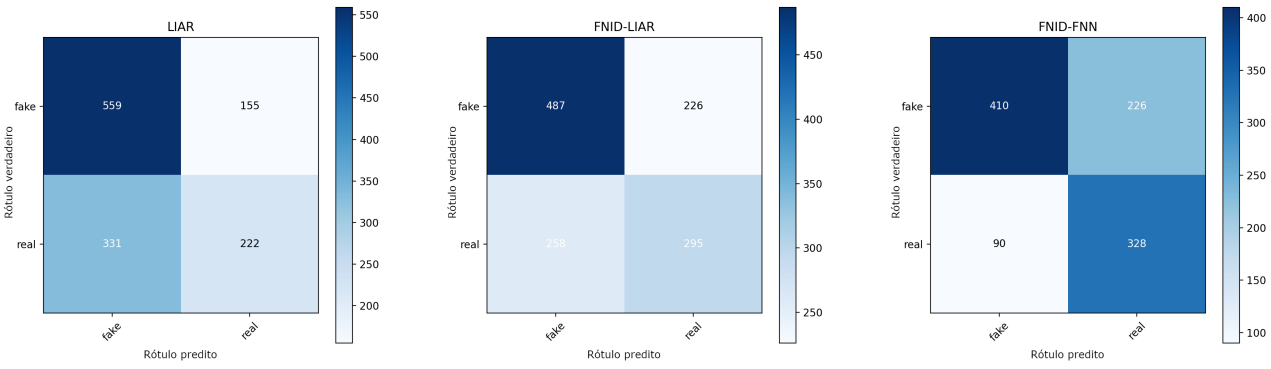
Fonte: Elaborado pelo autor.

Figura 10 – Matrizes de confusão para os modelos SS3.



Fonte: Elaborado pelo autor.

Figura 11 – Matrizes de confusão para os modelos CNN.



Fonte: Elaborado pelo autor.

no treinamento.

O conjunto FNID-FNN foi o melhor classificado em ambos modelos, com 0.6395 e 0.6850 de acurácia nos modelos SS3 e CNN, respectivamente. Mesmo sendo um modelo computacionalmente mais simples, o SS3 conseguiu uma performance comparável ao CNN nos conjuntos LIAR e LIAR-FNID.

4.5.1 Visualizando explicações

Com os modelos testados podemos aplicar as técnicas de explicação para visualização em algumas amostras do conjunto de teste. Em relação ao método de explicação LRP, foram testadas 4 regras aplicadas a rede CNN como um todo:

a) LRP-$\alpha\beta$ ($\alpha = 2, \beta = 1$);

b) LRP-ϵ ($\epsilon = 10^{-7}$);

c) LRP-z. Equivalente ao LRP-0;

d) LRP-z^+. Semelhante ao LRP-γ, mas só inclui as contribuições positivas.

Por conveniência, as amostras foram extraídas do conjunto de teste do FNID-FNN.

Com a Figura 12, é possível perceber diversos vieses que o modelo utiliza, especialmente pelo conjunto de dados (incluindo os outros utilizados nesse projeto) ter origem em declarações políticas. Verificando as regras que salientam contribuições positivas, o *token* `obama` tendeu a favorecer a predição da notícia pelo modelo como sendo falsa, visível em todas as regras de propagação utilizadas.

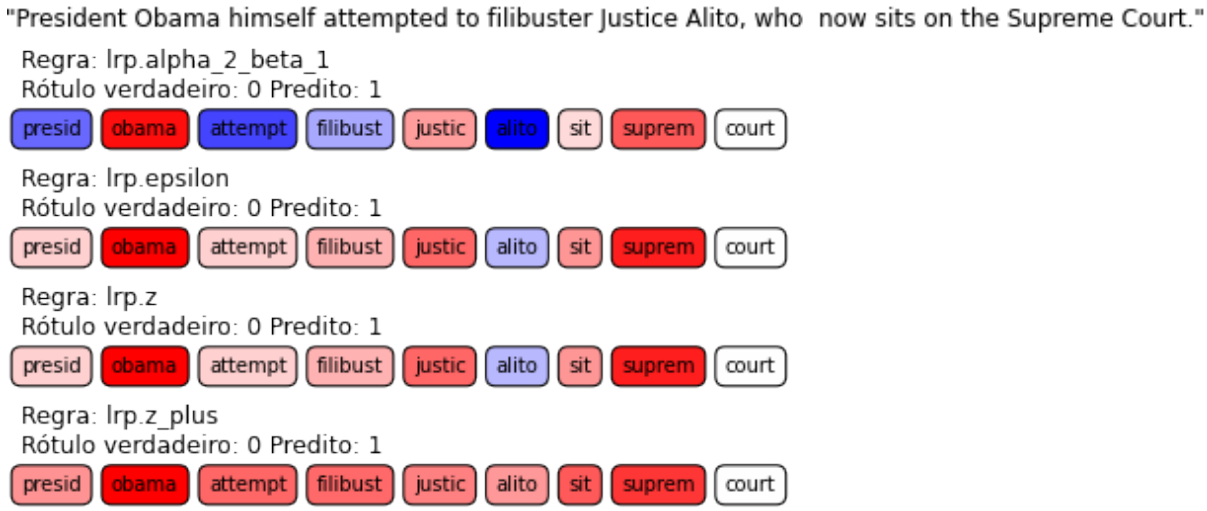
Do ponto de vista da interpretabilidade, a regra LRP-$\alpha\beta$ pode ser interessante de utilizar para verificar a tendencia do tipo de informação que contribui negativamente ou positivamente para a predição.

Na Figura 13, a regra LRP-$\alpha\beta$ verificou que o *token* `exceed` não contribui para a classificação dessa amostra, enquanto as regras LRP-ϵ/z destacaram o mesmo como uma contribuição positiva relevante. A falta de relevância no mesmo *token* pela regra LRP-z^+ corrobora com essa ideia.

Em ambos exemplos, é possível ver a redução em lexemas das palavras da frase original em cada célula.

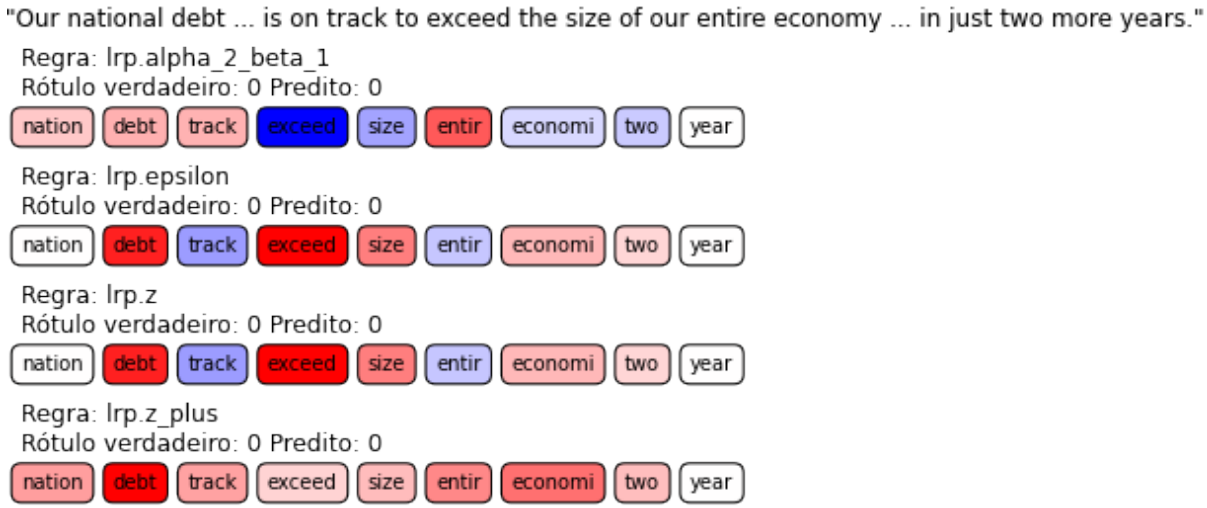
Observando as mesmas amostras com a explicação do modelo SS3, utilizando a ferramenta `Live_Test`, ambos modelos as classificaram com os mesmos rótulos. Na Figura 14 é possível ver o mesmo viés para o primeiro documento descrito anteriormente. Entretanto, é possível ver como o *token* `exceed` é utilizado para classificação em comparação aos dois rótulos possíveis. Diferentemente do que mostrado com LRP, ele foi utilizado para a classificação da amostra como `real`.

Figura 12 – Amostra com viés classificada incorretamente pelo modelo CNN explicada com LRP.



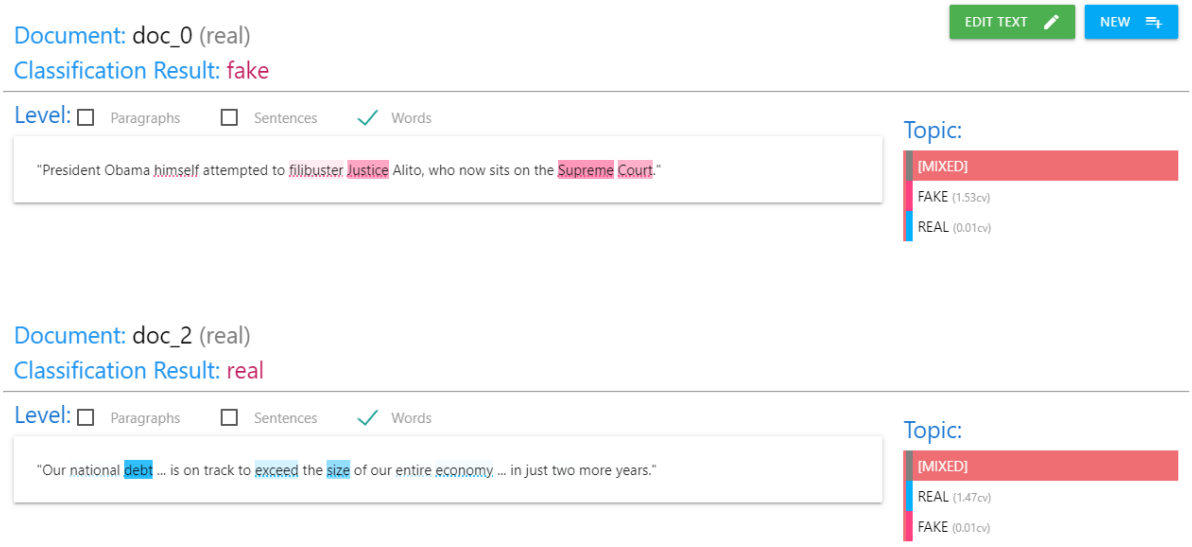
Fonte: Elaborado pelo autor.

Figura 13 – Amostra classificada corretamente pelo modelo CNN explicada com LRP.



Fonte: Elaborado pelo autor.

Figura 14 – Amostras classificadas pelo modelo SS3 explicada pelo mesmo.



Fonte: Elaborado pelo autor.

5 Conclusão

Durante esse trabalho foi possível enfrentar o desafio de classificação de desinformação com uma abordagem simples que obteve resultados razoáveis, em comparação com modelos de estado da arte, considerando a simplicidade dos modelos utilizados. Comparado a outras tarefas de Processamento de Linguagem Natural, acaba não sendo trivial especialmente se tratando somente do contexto do texto e utilizando modelos mais simples e diretos.

Foi possível analisar três conjuntos de dados semelhantes juntamente a dois modelos diferentes, contemplando uma grande quantidade de dados. Ambos modelos tiveram performances semelhantes, com o modelo CNN tendo uma pequena margem superior de performance, especialmente considerando o conjunto FNID-FNN, sendo que foi o conjunto de dados que obteve melhores métricas em ambos modelos, sendo a melhor uma acurácia de 0.6850.

Foi possível abordar e estudar os conceitos de explicabilidade e interpretabilidade dentro da área de Inteligência Artificial, considerando suas vantagens e melhorias que podem e deveriam ser adotadas em diversas aplicações e modelos utilizados no mundo atual, especialmente em casos em que decisões críticas devem ser tomadas.

Durante o desenvolvimento, foi possível estudar um modelo mais recente de classificação de texto que tem características interessantes e combina simplicidade e performance para certas tarefas, além de um método que explicabilidade que pode não só ser usado para abordar tarefas de Processamento de Linguagem Natural como vários outros problemas envolvendo modelos estruturados como redes neurais. Além disso, várias ferramentas foram estudadas, especialmente bibliotecas Python, possibilitando um entendimento do fluxo de trabalho de um pesquisador na área de Inteligência Artificial e de profissionais relacionados no mercado de trabalho.

Referências

ALBER, M.; LAPUSCHKIN, S.; SEEGERER, P.; HäGELE, M.; SCHüTT, K. T.; MONTAVON, G.; SAMEK, W.; MüLLER, K.-R.; DäHNE, S.; KINDERMANS, P.-J. iNNvestigate Neural Networks! *Journal of Machine Learning Research*, v. 20, n. 93, p. 1–8, 2019.

ALLCOTT, H.; GENTZKOW, M. Social media and fake news in the 2016 election. *Journal of economic perspectives*, v. 31, n. 2, p. 211–36, 2017.

AMIRKHANI, F. S. A. J. B. H. *FNID: Fake News Inference Dataset*. IEEE Dataport, 2020. Disponível em: <https://dx.doi.org/10.21227/fbzd-sw81>. Acesso em: 5 nov. 2020.

ARRAS, L.; HORN, F.; MONTAVON, G.; MÜLLER, K.-R.; SAMEK, W. Explaining predictions of non-linear classifiers in NLP. In: *Proceedings of the 1st Workshop on Representation Learning for NLP*. Berlin, Germany: Association for Computational Linguistics, 2016. p. 1–7.

ARRAS, L.; HORN, F.; MONTAVON, G.; MüLLER, K.-R.; SAMEK, W. "What is relevant in a text document?": An interpretable machine learning approach. *PLOS ONE*, Public Library of Science (PLoS), v. 12, n. 8, p. 1–23, ago 2017. ISSN 1932-6203.

ARRAS, L.; MONTAVON, G.; MÜLLER, K.-R.; SAMEK, W. Explaining recurrent neural network predictions in sentiment analysis. In: *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Copenhagen, Denmark: Association for Computational Linguistics, 2017. p. 159–168.

BACH, S.; BINDER, A.; MONTAVON, G.; KLAUSCHEN, F.; MÜLLER, K.-R.; SAMEK, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, Public Library of Science, v. 10, n. 7, 2015.

BELLE, V.; PAPANTONIS, I. Principles and practice of explainable machine learning. *arXiv preprint arXiv:2009.11698*, 2020.

BESSI, A.; FERRARA, E. Social bots distort the 2016 us presidential election online discussion. *First Monday*, v. 21, n. 11-7, 2016.

BURDISSO, S. G.; ERRECALDE, M.; GÓMEZ, M. Montes-y. Pyss3: A python package implementing a novel text classifier with visualization tools for explainable ai. *arXiv preprint arXiv:1912.09322*, 2019.

BURDISSO, S. G.; ERRECALDE, M.; GÓMEZ, M. Montes-y. A text classification framework for simple and effective early depression detection over social media streams. *Expert Systems with Applications*, Elsevier, v. 133, p. 182–197, 2019.

BURDISSO, S. G.; ERRECALDE, M.; GÓMEZ, M. Montes-y. τ-SS3: a text classifier with dynamic n-grams for early risk detection over text streams. *Pattern Recognition Letters*, Elsevier, v. 138, p. 130–137, 2020.

CHOLLET, F. et al. *Keras*. 2015. Disponível em: <https://keras.io>. Acesso em: 5 nov. 2020.

DOSHI-VELEZ, F.; KIM, B. Considerations for evaluation and generalization in interpretable machine learning. In: *Explainable and Interpretable Models in Computer Vision and Machine Learning*. [S.l.]: Springer, 2018. p. 3–17.

DOŠILOVIĆ, F. K.; BRČIĆ, M.; HLUPIĆ, N. Explainable artificial intelligence: A survey. In: IEEE. *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*. [S.l.], 2018. p. 0210–0215.

EGELHOFER, J. L.; AALDERING, L.; EBERL, J.-M.; GALYGA, S.; LECHELER, S. From novelty to normalization? how journalists use the term "fake news" in their reporting. *Journalism Studies*, Taylor & Francis, p. 1–21, 2020.

GENDREAU, H. *The Internet Made 'Fake News' a Thing—Then Made It Nothing*. 2017. Disponível em: <https://www.wired.com/2017/02/internet-made-fake-news-thing-made-nothing/>. Acesso em: 5 nov. 2020.

GILPIN, L. H.; BAU, D.; YUAN, B. Z.; BAJWA, A.; SPECTER, M.; KAGAL, L. Explaining explanations: An overview of interpretability of machine learning. In: IEEE. *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*. [S.l.], 2018. p. 80–89.

GRANIK, M.; MESYURA, V. Fake news detection using naive bayes classifier. In: IEEE. *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*. [S.l.], 2017. p. 900–903.

HARRIS, C. R.; MILLMAN, K. J.; WALT, S. J. van der; GOMMERS, R.; VIRTANEN, P.; COURNAPEAU, D.; WIESER, E.; TAYLOR, J.; BERG, S.; SMITH, N. J.; KERN, R.; PICUS, M.; HOYER, S.; KERKWIJK, M. H. van; BRETT, M.; HALDANE, A.; R'ıO, J. F. del; WIEBE, M.; PETERSON, P.; G'eRARD-MARCHANT, P.; SHEPPARD, K.; REDDY, T.; WECKESSER, W.; ABBASI, H.; GOHLKE, C.; OLIPHANT, T. E. Array programming with NumPy. *Nature*, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set 2020.

HUNTER, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007.

JOHNSON, R.; ZHANG, T. Effective use of word order for text categorization with convolutional neural networks. In: *HLT-NAACL*. [S.l.: s.n.], 2015.

KIM, Y. Convolutional neural networks for sentence classification. In: *EMNLP*. [S.l.: s.n.], 2014.

KLUYVER, T.; RAGAN-KELLEY, B.; PÉREZ, F.; GRANGER, B.; BUSSONNIER, M.; FREDERIC, J.; KELLEY, K.; HAMRICK, J.; GROUT, J.; CORLAY, S.; IVANOV, P.; AVILA, D.; ABDALLA, S.; WILLING, C.; TEAM, J. development. Jupyter notebooks - a publishing format for reproducible computational workflows. In: LOIZIDES, F.; SCMIDT, B. (Ed.). *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. Netherlands: IOS Press, 2016. p. 87–90.

LAZER, D. M.; BAUM, M. A.; BENKLER, Y.; BERINSKY, A. J.; GREENHILL, K. M.; MENCZER, F.; METZGER, M. J.; NYHAN, B.; PENNYCOOK, G.; ROTHSCHILD, D. et al. The science of fake news. *Science*, American Association for the Advancement of Science, v. 359, n. 6380, p. 1094–1096, 2018.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Ieee, v. 86, n. 11, p. 2278–2324, 1998.

LIU, Y.; WU, Y.-F. B. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In: *Thirty-second AAAI conference on artificial intelligence*. [S.l.: s.n.], 2018.

MCKINNEY Wes. Data Structures for Statistical Computing in Python. In: WALT Stéfan van der; MILLMAN Jarrod (Ed.). *Proceedings of the 9th Python in Science Conference*. [S.l.: s.n.], 2010. p. 56 – 61.

MCMILLAN, K.; WOOTSON JR., C. R. *Newseum pulls 'fake news' shirts after outcry from journalists*. 2018. Disponível em: <https://www.washingtonpost.com/news/arts-and-entertainment/wp/2018/08/03/the-newseum-is-selling-fake-news-shirts-journalists-are-not-amused/>. Acesso em: 5 nov 2020.

MOHSENI, S.; RAGAN, E.; HU, X. Open issues in combating fake news: Interpretability as an opportunity. *arXiv preprint arXiv:1904.03016*, 2019.

MONTAVON, G.; BINDER, A.; LAPUSCHKIN, S.; SAMEK, W.; MÜLLER, K.-R. Layer-wise relevance propagation: an overview. In: *Explainable AI: interpreting, explaining and visualizing deep learning*. [S.l.]: Springer, 2019. p. 193–209.

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

OTTER, D. W.; MEDINA, J. R.; KALITA, J. K. A survey of the usages of deep learning in natural language processing. *arXiv preprint arXiv:1807.10854*, 2018.

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

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

POERNER, N.; ROTH, B.; SCHÜTZE, H. Evaluating neural network explanation methods using hybrid documents and morphological agreement. *arXiv preprint arXiv:1801.06422*, 2018.

REIS, J. C. S.; CORREIA, A.; MURAI, F.; VELOSO, A.; BENEVENUTO, F. Explainable machine learning for fake news detection. In: *Proceedings of the 10th ACM Conference on Web Science*. New York, NY, USA: Association for Computing Machinery, 2019. p. 17–26.

RUCHANSKY, N.; SEO, S.; LIU, Y. Csi: A hybrid deep model for fake news detection. In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. [S.l.: s.n.], 2017. p. 797–806.

SAMEK, W.; WIEGAND, T.; MÜLLER, K.-R. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*, 2017.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural networks*, Elsevier, v. 61, p. 85–117, 2015.

SHRIKUMAR, A.; GREENSIDE, P.; KUNDAJE, A. Learning important features through propagating activation differences. *CoRR*, abs/1704.02685, 2017.

SHU, K.; CUI, L.; WANG, S.; LEE, D.; LIU, H. defend: Explainable fake news detection. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. [S.l.: s.n.], 2019. p. 395–405.

SHU, K.; MAHUDESWARAN, D.; WANG, S.; LEE, D.; LIU, H. Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media. *arXiv preprint arXiv:1809.01286*, 2018.

WANG, W. Y. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. [S.l.]: Association for Computational Linguistics, 2017. p. 422–426.

YANG, F.; PENTYALA, S. K.; MOHSENI, S.; DU, M.; YUAN, H.; LINDER, R.; RAGAN, E. D.; JI, S.; HU, X. Xfake: explainable fake news detector with visualizations. In: *The World Wide Web Conference*. [S.l.: s.n.], 2019. p. 3600–3604.