UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS CRATEÚS

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SAULO CAVALCANTE DA SILVA

DETECÇÃO AUTOMATIZADA DE DESINFORMADORES EM GRUPOS PÚBLICOS DO WHATSAPP UTILIZANDO REDES NEURAIS PARA GRAFOS

CRATEÚS

2025

SAULO CAVALCANTE DA SILVA

DETECÇÃO AUTOMATIZADA DE DESINFORMADORES EM GRUPOS PÚBLICOS DO WHATSAPP UTILIZANDO REDES NEURAIS PARA GRAFOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Crateús da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Dr. Jose Wellington Franco da Silva.
Coorientador: Prof. Dr. Rennan Ferreira Dantas.

CRATEÚS

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S583d Silva, Saulo Cavalcante da.
Detecção automatizada de desinformadores em grupos públicos do whatsApp utilizando redes neurais para grafos / Saulo Cavalcante da Silva. – 2025.
82 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Crateús, Curso de Ciência da Computação, Crateús, 2025.
Orientação: Prof. Dr. Jose Wellington Franco da Silva.
Coorientação: Prof. Dr. Rennan Ferreira Dantas.

1. desinformação. 2. detecção de desinformadores. 3. redes neurais para grafos. 4. whatsApp. I. Título.
CDD 004

SAULO CAVALCANTE DA SILVA

DETECÇÃO AUTOMATIZADA DE DESINFORMADORES EM GRUPOS PÚBLICOS DO WHATSAPP UTILIZANDO REDES NEURAIS PARA GRAFOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Crateús da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em: xx/xx/xxxx.

BANCA EXAMINADORA

Prof. Dr. Jose Wellington Franco da Silva (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Rennan Ferreira Dantas (Coorientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Rafael Martins Barros
Universidade Federal do Ceará (UFC)

Prof. Dr. Marciel Barros Pereira
Universidade Federal do Ceará (UFC)

Dedico este trabalho à minha família, em especial à minha mãe e ao meu pai.

AGRADECIMENTOS

À Universidade Federal do Ceará, pela oportunidade de realizar minha graduação e por ter sido meu lar ao longo de toda essa jornada.

À minha família, em especial aos meus pais, Silvestre Albino e Maria Silvana, por todo o amor, carinho, apoio e por sempre se esforçarem em me guiar pelos caminhos corretos.

Agradeço ao meu orientador, Prof. Dr. Wellington Franco, e ao meu coorientador, Prof. Dr. Rennan Dantas, por todo o apoio prestado e por nunca terem desistido de mim.

À banca examinadora, por sua atenção cuidadosa, pelas relevantes observações e pelo valioso tempo dedicado à análise deste trabalho, que contribuíram significativamente para seu aprimoramento.

Aos professores que me acompanharam e, com paciência, me aturaram desde o ensino fundamental até o médio, e a todos os docentes que contribuíram para a minha trajetória acadêmica, meu sincero agradecimento.

A todas as pessoas envolvidas no restaurante universitário, pelo excelente trabalho realizado diariamente.

Aos colegas e amigos que estiveram ao meu lado ao longo desta jornada, nos desafios acadêmicos, nos momentos de descontração, nas alegrias e nas adversidades, minha sincera gratidão por tornarem essa trajetória mais leve e divertida.

Estendo meus agradecimentos a todos aqueles que, direta ou indiretamente, dentro ou fora da universidade, contribuíram para minha formação ao longo dessa jornada.

"Se o Pica-Pau tivesse comunicado à polícia,
isso nunca teria acontecido!"

(Captain Haddock - Pica Pau)

RESUMO

Nos últimos anos, a disseminação de informações falsas em plataformas digitais tem gerado crescentes preocupações sociais, especialmente no que diz respeito ao seu impacto sobre a opinião pública, os processos democráticos e a saúde coletiva. Dentre essas plataformas, o *WhatsApp* se destaca no contexto brasileiro como um dos principais meios de propagação de conteúdos desinformativos, sobretudo em grupos públicos de grande alcance. Embora a literatura sobre detecção de desinformação venha avançando, ainda são escassos os estudos voltados especificamente à identificação de desinformadores nesse ambiente. Este trabalho propõe uma abordagem inovadora para a detecção automatizada desses agentes, utilizando Graph Neural Networks (GNN) aplicadas a um conjunto real de dados coletados de grupos públicos do *WhatsApp*. A metodologia envolve a modelagem das interações entre usuários na forma de um grafo, permitindo capturar aspectos estruturais e dinâmicos do comportamento de disseminação. Por meio de uma análise comparativa entre diferentes arquiteturas de GNN: (Graph Convolutional Network (GCN), Graph Convolutional Network II (GCNII), Graph Sample and Aggregation (GraphSAGE), Graph Attention Network (GAT) e Approximate Personalized Propagation of Neural Predictions (APPNP)), avaliou-se a eficácia das abordagens na tarefa de identificar desinformadores. Os resultados obtidos confirmam o potencial das GNN em capturar padrões semânticos e topológicos, contribuindo para o aprimoramento das estratégias de enfrentamento à desinformação no *WhatsApp* e consolidando essa abordagem no domínio.

Palavras-chave: desinformação; detecção de desinformadores; redes neurais para grafos; whatsApp.

ABSTRACT

In recent years, the spread of false information on digital platforms has raised growing social concerns, particularly regarding its impact on public opinion, democratic processes, and public health. Among these platforms, WhatsApp stands out in the brazilian context as one of the main channels for the dissemination of disinformation, especially within large public groups. Although the literature on disinformation detection has made significant progress, studies specifically focused on identifying misinformers in this environment remain scarce. This work proposes an innovative approach for the automated detection of such agents, leveraging GNN applied to a real dataset collected from public WhatsApp groups. The methodology involves modeling user interactions as a graph, enabling the capture of structural and dynamic aspects of dissemination behavior. Through a comparative analysis of different GNN architectures: (GCN, GCNII, GraphSAGE, GAT e APPNP), the effectiveness of these approaches in identifying misinformers was evaluated. The results confirm the potential of GNN to capture both semantic and topological patterns, thereby strengthening disinformation mitigation strategies on *WhatsApp* and further establishing this approach in the field.

Keywords: disinformation; misinformer detection; graph neural networks; whatsApp.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS-FONTE

LISTA DE ABREVIATURAS E SIGLAS

APIs	Application Programming Interfaces
APPNP	Approximate Personalized Propagation of Neural Predictions
COVID-19	Coronavirus Disease 2019
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GCNII	Graph Convolutional Network II
GNN	Graph Neural Networks
GraphSAGE	Graph Sample and Aggregation
IA	Inteligência Artificial
LINE	Large-scale Information Network Embedding
PLN	Processamento de Linguagem Natural
PPR	Personalized PageRank
ReLU	Rectified Linear Unit
TF-IDF	Term Frequency-Inverse Document Frequency

SUMÁRIO

1 INTRODUÇÃO

De acordo com Metzler e Garcia (2024), as redes sociais digitais têm promovido uma transformação profunda nos processos informacionais, ao proporcionarem acesso quase instantâneo e de baixo custo a uma ampla variedade de conteúdos. Sua arquitetura algorítmica, baseada em mecanismos de recomendação e personalização, intensifica a viralização das publicações, ampliando exponencialmente o alcance e a velocidade com que as informações se disseminam.

Esse caráter descentralizado, dinâmico e pouco regulado também facilita a disseminação da desinformação, uma vez que permite a publicação de conteúdos sem critérios rigorosos de veracidade ou responsabilidade (FERREIRA *et al.*, 2021b). Nesse cenário, destacam-se os desinformadores, definidos por Su *et al.* (2020) como usuários maliciosos que, de forma intencional e recorrente, propagam desinformação em larga escala, distinguindo-se pelo alto volume de compartilhamento de conteúdos falsos ou enganosos.

Diante da crescente proliferação de conteúdos falsos e manipuladores nas redes sociais, a identificação e o combate a agentes desinformadores tornaram-se temas centrais em diversas pesquisas acadêmicas. Trabalhos recentes como DeVerna *et al.* (2024), Khan *et al.* (2023) e Rath *et al.* (2022) têm se dedicado à construção de modelos preditivos e à análise comportamental de usuários para otimizar a detecção de desinformadores em ambientes digitais. Entre as plataformas sociais, o *X* (anteriormente conhecido como Twitter) é o principal foco de investigação por sua grande base de usuários, rapidez na disseminação de informações e *Application Programming Interfaces (APIs)* públicas que facilitam o acesso à dados estruturados e históricos de interações.

Porém, no Brasil, o *WhatsApp* constitui a principal rede social de comunicação do país. Segundo a matéria BP MONEY (2025), o Brasil registra aproximadamente 197 milhões de usuários ativos da aplicação, posicionando-se como o segundo maior mercado mundial do serviço, superado apenas pela Índia. Ademais, de acordo com levantamento realizado pela Câmara dos Deputados e pelo Senado, que ouviu 2,4 mil pessoas, 79% dos brasileiros recorrem ao *WhatsApp* como principal fonte de informação (Rádio Senado, 2019). Entretanto, essa penetração expressiva tem favorecido a propagação de desinformação, especialmente durante períodos eleitorais e em momentos de tensão social.

Segundo a pesquisa Fiocruz (2020), 73,7% do conteúdo desinformativo compartilhado acerca da Coronavirus Disease 2019 (COVID-19) foi disseminado via *WhatsApp*. Ademais,

conforme apontado por Ferreira *et al.* (2021a), a disseminação de conteúdos desinformativos sobre efeitos adversos e eficácia vacinal contribuiu substancialmente para o fenômeno da hesitação vacinal, corroborando os efeitos deletérios da desinformação na saúde pública.

Nesse contexto, o combate à desinformação no *WhatsApp* configura-se como uma demanda premente, especialmente considerando o papel central que a plataforma exerce na dinâmica comunicacional da sociedade brasileira. Pesquisas como a de Cabral *et al.* (2023) têm se dedicado à identificação de conteúdos desinformativos e de seus disseminadores, por meio da coleta de dados em grupos públicos e da construção de conjuntos de dados específicos para esse fim. Contudo, o referido trabalho concentra-se majoritariamente na detecção de desinformação, e não na identificação dos agentes desinformadores.

Diante do cenário em análise, o presente estudo propõe uma metodologia aprofundada para a identificação de disseminadores de desinformação em grupos públicos do *WhatsApp*, empregando uma combinação de técnicas avançadas de GNN. Para tanto, são investigados cinco modelos distintos (GCN, GCNII, GraphSAGE, GAT e APPNP), os quais apresentam especificidades que possibilitam a captura das complexas dinâmicas e interações entre os usuários. A abordagem baseia-se na construção de um grafo representativo das interações, em que os nós correspondem aos usuários e as arestas são estabelecidas a partir da participação conjunta em grupos, considerando, para sua ponderação, tanto a quantidade de mensagens quanto o grau de viralização. Este modelo de representação permite a extração de características do comportamento dos usuários, evidenciando padrões que podem indicar a propagação de desinformação.

O estudo utiliza métodos de preparação dos dados, segmentando-os em conjuntos de treinamento, validação e teste por meio de validação cruzada estratificada, o que assegura a representatividade e reduz vieses. Os modelos de GNN são treinados com técnicas de regularização, normalização e *early stopping*, e avaliados por métricas como acurácia, precisão, *recall* e *F1-score*. Dessa forma, a abordagem comparativa proposta não só avalia a eficácia dos modelos na detecção de disseminadores de desinformação, mas também contribui para o aprofundamento da compreensão dos mecanismos subjacentes à propagação de informações falsas em ambientes digitais, em especial no *WhatsApp*.

1.1 Justificativa

A identificação de desinformadores é crucial para salvaguardar a integridade dos processos democráticos, a saúde pública e a estabilidade social, especialmente em contextos

onde o *WhatsApp* é a principal fonte de informação. A complexidade inerente à disseminação de conteúdos falsos, associada ao caráter criptografado e privado dos grupos no aplicativo, impõe desafios significativos para as abordagens tradicionais (CABRAL *et al.*, 2023). Nesse cenário, a utilização de técnicas avançadas de GNN revela-se uma estratégia inovadora e promissora para superar as limitações dos métodos convencionais. Tais modelos possibilitam uma análise aprofundada das interações entre usuários, integrando informações sobre a topologia das conexões e as características individuais em um mesmo *framework*. Dessa forma, este trabalho justifica-se pelo seu potencial de contribuir para o avanço do conhecimento na detecção de desinformadores, ampliando as possibilidades de aplicação prática de técnicas de aprendizado profundo em ambientes digitais altamente dinâmicos e complexos.

1.2 Objetivo Geral

Identificar e classificar agentes disseminadores de desinformação na plataforma *WhatsApp*, por meio da aplicação de técnicas de redes neurais para grafos, de forma a aferir sistematicamente a eficácia relativa de cada método na detecção desses agentes em grupos públicos da plataforma.

1.3 Objetivos Específicos

- Realizar a coleta e o pré-processamento do conjunto de dados *FakeWhatsApp.Br*;
- Elaborar uma modelagem das interações em grupos públicos do *WhatsApp* por meio de uma representação em formato de grafo;
- Desenvolver os modelos GNN, incorporando técnicas de regularização, normalização e *early stopping*;
- Dividir o conjunto de dados em subconjuntos de treinamento, validação e teste por meio de validação cruzada estratificada;
- Monitorar o desempenho dos modelos por meio de métricas quantitativas como acurácia, precisão, *recall* e *F1-score*;
- Analisar e comparar os resultados obtidos com os diferentes modelos, identificando as vantagens, limitações e potenciais aplicações práticas de cada abordagem.

1.4 Organização do Trabalho

Esta pesquisa está estruturada em seis capítulos, os quais se articulam de forma integrada para fundamentar e desenvolver os métodos propostos para a identificação de disseminadores de desinformação no *WhatsApp*.

- No Capítulo 2, a Fundamentação Teórica apresenta os conceitos centrais de desinformação, teoria dos grafos e redes neurais, com ênfase em modelos GNN;
- O Capítulo 3 realiza uma análise crítica da literatura, destacando estudos anteriores sobre identificação de disseminadores de desinformação e modelagem de interações em redes sociais;
- No Capítulo 4, detalha-se a Metodologia: coleta e pré-processamento dos dados, construção do grafo de usuários e implementação dos modelos de GNN;
- O Capítulo 5 expõe os Resultados e a Análise Experimental, comparando o desempenho das abordagens e apontando suas vantagens e limitações;
- Por fim, o Capítulo 6 sintetiza as conclusões e sugere direções para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, desenvolve-se a base teórica necessária para a compreensão dos conceitos abordados ao longo deste trabalho. Inicialmente, são abordadas definições de desinformação e desinformadores, bem como as diversas formas de manifestação desses conceitos nas redes sociais. Além disso, introduzem-se os principais fundamentos da Teoria dos Grafos, que são essenciais para a modelagem das interações entre usuários em plataformas de redes sociais. Por fim, são trazidas noções de Inteligência Artificial e de Redes Neurais, com enfoque nas redes neurais para grafos.

2.1 Desinformação

Conforme Su *et al.* (2020), desinformação refere-se à disseminação regularmente intencional de informações deturpadas, englobando conteúdos falsos, enganosos, forjados, imprecisos, descontextualizados ou distorcidos, frequentemente elaborados com o propósito deliberado de atingir objetivos específicos. Dessa forma, a desinformação distingue-se de outras formas de informações incorretas por sua natureza ordinariamente intencional, sendo criada com o objetivo de causar prejuízos diversos ou influenciar o comportamento de audiências em benefício de determinados interesses. Essa definição abarca uma ampla variedade de práticas, desde a produção de notícias falsas até a manipulação seletiva de fatos.

Embora anteceda a era digital, a desinformação passou a circular com muito mais velocidade, capilaridade e volume após a popularização da internet e das redes sociais. O fácil acesso a essas plataformas, aliado ao seu baixo custo ou gratuidade, descentralizou o fluxo de informações, reduzindo a dependência das mídias convencionais e permitindo que qualquer usuário torne-se um potencial criador e disseminador de conteúdo. Como apontam Vosoughi *et al.* (2018), as redes sociais não apenas facilitaram a disseminação da desinformação, mas também a amplificaram de maneira sem precedentes, desafiando a capacidade da sociedade de distinguir entre informações verdadeiras e falsas.

Diferentemente da desinformação em meios tradicionais, nas redes sociais, o conteúdo pode ser criado e compartilhado por qualquer usuário, sem a necessidade de verificação prévia, o que contribui para a rápida propagação de informações duvidosas. Conforme observado por Bakir e McStay (2018), os algoritmos das plataformas de mídia social amplificam a desinformação ao promoverem conteúdos sensacionalistas que, embora falsos, são mais propensos a

atrair cliques e compartilhamentos, resultando em uma propagação mais rápida e abrangente de notícias falsas em comparação com informações verídicas. Logo, tais algoritmos são projetados para maximizar o engajamento dos usuários, priorizando conteúdos que geram maiores níveis de interação, independentemente de sua veracidade.

A definição de desinformação apresentada por Su *et al.* (2020) é bastante abrangente. Assim, no contexto das redes, podemos nos deparar com diferentes formas de representações de conteúdos enganosos. A seguir, será realizado uma breve exposição das manifestações mais comuns de desinformação nas plataformas digitais.

2.1.1 Enganação

Enganação envolve práticas intencionais destinadas a levar as pessoas a acreditar em algo que não é verdadeiro. Como destacado por Zhou *et al.* (2004), trata-se da ação deliberada de fornecer informações falsas ou enganosas, sendo uma tática amplamente utilizada em contextos nos quais o manipulador visa influenciar percepções e decisões alheias, frequentemente em proveito próprio. Essas práticas incluem métodos como a omissão de informações, a distorção de fatos e a manipulação de conteúdos, resultando em percepções distorcidas da realidade.

2.1.2 Conteúdo Manipulado

Conforme apontado por Chesney e Citron (2019), conteúdo manipulado constitui uma forma de desinformação na qual a tecnologia é empregada para criar falsificações altamente convincentes, alterando imagens, vídeos ou áudios de modo que as pessoas retratadas aparentem realizar ações ou proferir declarações que, de fato, nunca ocorreram. Ademais, verifica-se que as tecnologias de falsificação tornam-se cada vez mais acessíveis, facultando a indivíduos desprovidos de conhecimentos técnicos avançados a capacidade de produzir conteúdos forjados com elevado grau de sofisticação.

2.1.3 Fake News

Conforme a definição proposta por Jr *et al.* (2018), as *fake news* consistem em informações confeccionadas de forma deliberada e disseminadas com o intuito de enganar o público, apresentando-se frequentemente sob o formato de reportagens legítimas, mas desprovidas de embasamento factual ou distorcidas para atender a interesses específicos. Essa modalidade

de desinformação revela-se particularmente perniciosa, na medida em que explora a credibilidade inerente ao jornalismo, distorcendo a percepção pública e conduzindo-a a conclusões equivocadas.

2.1.4 Teorias da Conspiração

Teorias da conspiração são narrativas que sugerem explicações alternativas para eventos ou situações, frequentemente fundamentadas em suspeitas de má-fé por parte de instituições ou grupos influentes (LANDRUM; OLSHANSKY, 2019). Como apontado por Wood *et al.* (2012), as teorias da conspiração são frequentemente impulsionadas pela desconfiança em relação a autoridades estabelecidas e tendem a se propagar rapidamente em comunidades marcadas por um ceticismo generalizado em relação às explicações oficiais. Esses discursos podem exercer um impacto profundo sobre a opinião pública, alimentando divisões sociais e enfraquecendo a confiança nas instituições.

2.1.5 Rumores

Rumores em redes sociais referem-se a informações não verificadas que se espalham rapidamente através de plataformas digitais, frequentemente impulsionadas por incertezas, medos ou curiosidades do público (BORDIA; DIFONZO, 2004). Esses rumores podem ser verdadeiros, falsos ou uma combinação de ambos, e sua propagação é amplificada pela natureza viral das redes sociais, onde o compartilhamento de informações ocorre de maneira ágil e sem a necessidade de verificação prévia.

A Tabela 1 apresenta uma síntese das diferentes manifestações de desinformação no contexto das redes sociais, fornecendo, para cada categoria, uma breve definição e as principais práticas ou características associadas.

2.2 Desinformadores

As redes sociais, ao democratizarem a criação e disseminação de conteúdo, também propiciaram a proliferação de informações falsas ou enganosas, frequentemente impulsionadas por atores que se engajam no constante compartilhamento de conteúdos questionáveis. Esses indivíduos ou grupos tornaram-se figuras centrais nas dinâmicas de desinformação online, valendo-se das plataformas digitais para alcançar vastas audiências com conteúdos que, em

Tabela 1 – Síntese das Formas de Desinformação

Categoria	**Descrição**	**Práticas / Características**
Enganação	Ação deliberada de fornecer informações falsas ou enganosas com o objetivo de influenciar percepções e decisões.	Omissão de dados, distorção de fatos, manipulação de conteúdo.
Conteúdo Manipulado	Uso de tecnologia para criar falsificações convincentes em imagens, vídeos ou áudios.	*Deepfakes*, edição sofisticada de mídia.
fake news	Informações criadas deliberadamente em formato de reportagem legítima, porém sem base factual.	Titulação sensacionalista, falta de fontes confiáveis.
Teorias da Conspiração	Narrativas que oferecem explicações alternativas para eventos, rejeitando evidências contrárias.	Desconfiança institucional, adesão comunitária.
Rumores	Informações não verificadas que se espalham em redes sociais, podendo ser verdadeiras, falsas ou mistas.	Propagação viral, ausência de verificação.

Fonte: O autor

muitos casos, distorcem a realidade. Conforme Su *et al.* (2020), a descentralização intrínseca às plataformas sociais confere a qualquer usuário, intencionalmente ou não, o potencial de atuar como vetor de desinformação, intensificando os desafios de monitoramento e verificação da veracidade dos conteúdos.

A produção de desinformação configura-se, em sua origem, como ato intencional; não obstante, o compartilhamento desse material muitas vezes ocorre sem motivação malévola consciente. Porém, indivíduos que compartilham desinformação de forma não intencional raramente se configuram como vetores primários de difusão em larga escala (SU *et al.*, 2020). Em contrapartida, os vetores principais da propagação maliciosa de conteúdos enganosos são constituídos por agentes que, de modo deliberado e estratégico, visam manipular a percepção pública e alcançar amplas audiências (SU *et al.*, 2020).

A definição de um usuário como desinformador pode variar conforme a rede social ou o comportamento específico que se busca identificar. Adotaremos, portanto, a definição abrangente trazida por Chu *et al.* (2010), segundo a qual um desinformador é aquele que exibe um comportamento anômalo em relação aos demais usuários, caracterizado não apenas pela criação e disseminação de informações falsas ou enganosas, mas também pelo uso estratégico das funcionalidades das redes sociais, como a viralidade e a personalização de conteúdo, para manipular percepções e influenciar comportamentos de maneira amplificada e direcionada.

2.2.1 Categorias de desinformadores

No âmbito das redes sociais, é possível distinguir diversos perfis de agentes responsáveis pela desinformação, cada um dotado de motivações e características específicas, mas todos convergindo para a propagação de conteúdo falso ou enganoso. Dentre essas categorias, destacam-se os *bots*, propagandistas e usuários comuns desinformados.

2.2.1.1 Bots

Bots constituem agentes automatizados programados para interagir em plataformas de redes sociais, simulando padrões de comportamento típicos de usuários humanos (FERRARA *et al.*, 2016). Tais programas podem ser configurados para executar, de forma massiva e contínua, ações como publicação de postagens, "curtidas" e compartilhamentos, o que lhes confere elevada capacidade de amplificação de conteúdo sem necessidade de intervenção direta de operadores humanos. Conforme observado por Ferrara *et al.* (2016), esses agentes desempenham papel central nas dinâmicas de engajamento das redes sociais, respondendo por parcela significativa das interações registradas.

Uma variação dos *bots* tradicionais são os chamados *cyborgs*, que combinam elementos de automação com a intervenção humana. Conforme Woolley e Howard (2016), essas contas híbridas executam atividades de larga escala, tais como postagens e compartilhamentos massivos por meio de algoritmos, ao mesmo tempo em que operadores humanos promovem ajustes contextuais e refinam interações mais sofisticadas.

2.2.1.2 Propagandistas

Segundo Jowett e O'donnell (2018), a propaganda constitui, em sua essência, um esforço intencional para moldar percepções, manipular cognições e direcionar comportamentos, a fim de gerar uma resposta alinhada com os interesses do propagandista. Embora, em contextos tradicionais, seja predominantemente empregada para promover produtos ou difundir ideias sem caráter malicioso ou enganoso, nas redes sociais ela frequentemente se articula com estratégias de desinformação, ampliando sua eficácia na produção de efeitos específicos e dirigidos.

Conforme Jowett e O'donnell (2018), o poder da propaganda decorre de sua habilidade em mobilizar vieses cognitivos e afetivos preexistentes, por meio de narrativas resumidas ou distorcidas que atendem aos interesses de seus promotores. Dessa combinação de estímulos

emocionais e apelos racionais resulta um instrumento de persuasão capaz de consolidar discursos hegemônicos e sustentar agendas políticas ou ideológicas específicas.

2.2.1.3 Usuários comuns desinformados

Conforme Pennycook *et al.* (2020), indivíduos comuns, sejam eles crédulos ou não, frequentemente, mesmo sem motivações maliciosas, operam como vetores significativos na proliferação de desinformação ao disseminarem conteúdos virais sem procedência verificada. Esse fenômeno se manifesta, especialmente, entre usuários altamente ativos, que compartilham grandes quantidades de conteúdo viral. Conforme será abordado nas seções subsequentes deste trabalho, os conteúdos virais têm uma probabilidade substancialmente elevada de conter, ou serem inteiramente compostos por, desinformação.

A Tabela 2 consolida uma visão sintética das distintas categorias de agentes de propagação de desinformação no âmbito das redes sociais, apresentando, para cada categoria, uma definição concisa e os principais atributos e práticas características.

Tabela 2 – Síntese das categorias de desinformadores em Redes Sociais

Categoria	**Descrição**	**Práticas / Características**
Bots	Agentes totalmente automatizados programados para interagir simulando usuários humanos.	Postagens, "curtidas" e compartilhamentos massivos e contínuos sem intervenção humana.
Cyborgs	Contas híbridas que unem automação de larga escala com intervenção humana para ajustes contextuais.	Publicações e compartilhamentos automatizados combinados com refinamento manual de interações.
Propagandistas	Agentes que moldam percepções e comportamentos por meio de narrativas distorcidas e viés cognitivo.	Omissão de dados, distorção de fatos, apelos emocionais e racionais para sustentar agendas.
Usuários comuns desinformados	Indivíduos que endossam e replicam narrativas falsas sem verificação de fontes, atuando como vetores.	Compartilhamento viral sem checagem de procedência, influenciados por viés de confirmação.

Fonte: O autor

2.2.2 Definição formal de detecção de desinformadores

A identificação de agentes responsáveis pela disseminação de desinformação em plataformas de redes sociais consolidou-se como um campo de estudo de elevada relevância, dado que a propagação de conteúdos falsos ou enganosos acarreta riscos substanciais à integridade do debate público e à estabilidade dos regimes democráticos (SHU *et al.*, 2017). Segundo Shu *et al.* (2017), o reconhecimento de agentes de desinformação baseia-se na aplicação de técnicas sofisticadas de análise de dados, métodos de aprendizado de máquina e procedimentos de verificação fática, com o propósito de revelar padrões comportamentais atípicos e identificar a propagação de conteúdos enganosos.

A base de dados utilizada neste trabalho provém do estudo de Cabral *et al.* (2023), que investigou a propagação de desinformação no *WhatsApp*. Portanto, este trabalho adotará a mesma definição formal de detecção de desinformadores proposta no estudo supracitado, garantindo a consistência metodológica e a comparabilidade dos resultados.

A formulação do problema de detecção de agentes de desinformação varia na literatura em função do enfoque adotado pelos autores (por exemplo, quando o foco recai na identificação de *bots* ou de propagandistas humanos). Neste trabalho, propomos uma definição abrangente, apresentada a seguir.

Seja um usuário $u = (U_u, E_u)$ de uma rede social N, que possui associado a ele um conjunto $U_u = \{u_1, u_2, \ldots, u_m\}$ de outros usuários com os quais ele possui uma conexão, e um conjunto de engajamentos $E_u = \{e_{u_1}, e_{u_2}, \ldots, e_{u_n}\}$, onde cada $e_{u_i} = (p_i, a, t)$ representa um engajamento de u com a publicação p_i, pela ação a, no tempo t. Seja a função $Q(s) \in [0,1]$ um score de desinformador atribuído a u e τ um limiar de decisão. A detecção dos desinformadores é a tarefa de aprender uma função de predição $G(u) \in \{0,1\}$, satisfazendo:

$$G(u) = \begin{cases} 1 \text{ (é desinformador)}, & \text{se } Q(s) \geq \tau \\ 0 \text{ (não é desinformador)}, & \text{se } Q(s) < \tau \end{cases} \tag{2.1}$$

A partir dessa definição abrangente, torna-se factível identificar, no ambiente do *WhatsApp*, os usuários desinformadores por meio da aplicação de técnicas de análise de rede social e algoritmos de aprendizado de máquina voltados à detecção de padrões comportamentais atípicos. Uma vez mapeados tais agentes de desinformação, podem-se implementar estratégias automatizadas de contenção dos mesmos.

2.3 Teoria dos grafos

Em conformidade com Bondy e Murty (2008), a Teoria dos Grafos é um ramo da matemática e da ciência da computação que se dedica ao estudo de grafos, estruturas formadas por vértices e arestas que conectam pares de vértices. Abaixo seguem os conceitos fundamentais relacionados à Teoria dos Grafos com base nas definições encontradas nos livros de West *et al.* (2001) e Bondy *et al.* (1976).

2.3.1 Conceitos Fundamentais

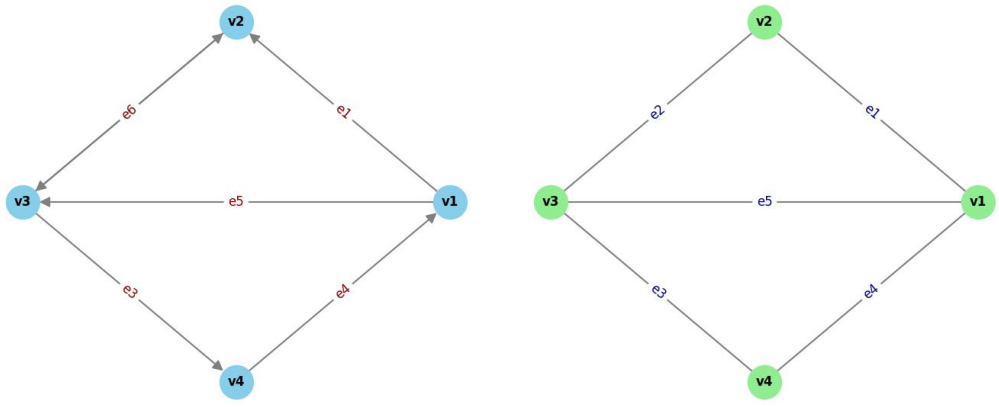
Um grafo G é definido como um par $G = (V, E)$, onde: V é um conjunto não vazio de vértices, também chamados de nós ou pontos; E é um conjunto de pares de vértices, chamados de arestas ou ligações. Afirma-se que o grafo G é finito se, e somente se, ambos os conjuntos V e E apresentam cardinalidade finita.

2.3.1.1 Grafos Direcionados e Não Direcionados

- **Grafo Direcionado (Digrafo):** Cada aresta tem uma direção, apontando de um vértice para outro.
- **Grafo Não Direcionado:** As arestas não possuem direção, ou seja, a relação entre os vértices é bidirecional.

Na Figura 1, ilustram-se dois grafos definidos sobre o mesmo conjunto de vértices $V = \{v_1, v_2, v_3, v_4\}$: o grafo à esquerda é direcionado, apresentando cinco arestas direcionadas, ao passo que o grafo à direita é não direcionado, composto por cinco arestas.

Figura 1 – Exemplo de grafo direcionado (à esquerda) e grafo não direcionado (à direita).



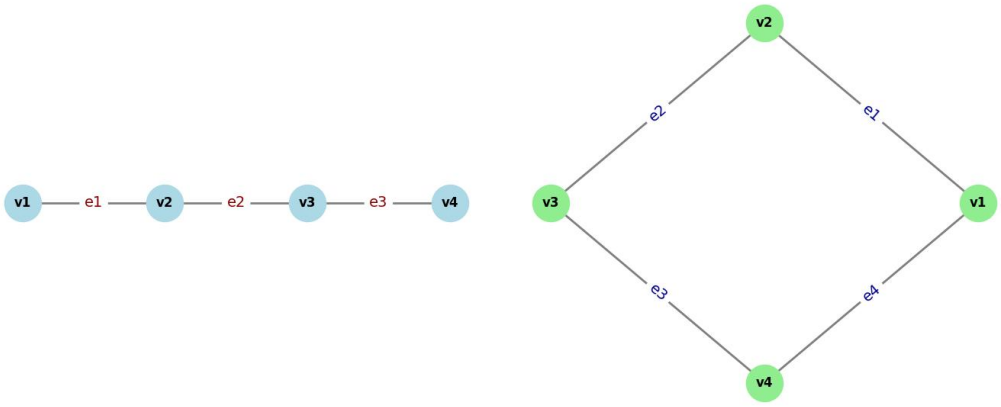
Fonte: O autor

2.3.1.2 *Caminho e ciclo*

- **Caminho:** Uma sequência de vértices conectados por arestas, onde cada vértice é visitado apenas uma vez.
- **Ciclo:** Um caminho que começa e termina no mesmo vértice, sem repetir arestas ou vértices.

Na Figura 2, apresentam-se dois grafos definidos sobre o mesmo conjunto de vértices $V = \{v_1, v_2, v_3, v_4\}$: à esquerda, um caminho composto por três arestas; à direita, um ciclo composto por quatro arestas.

Figura 2 – Exemplo de caminho (à esquerda) e de ciclo (à direita).



Fonte: Próprio autor

2.3.1.3 *Conectividade*

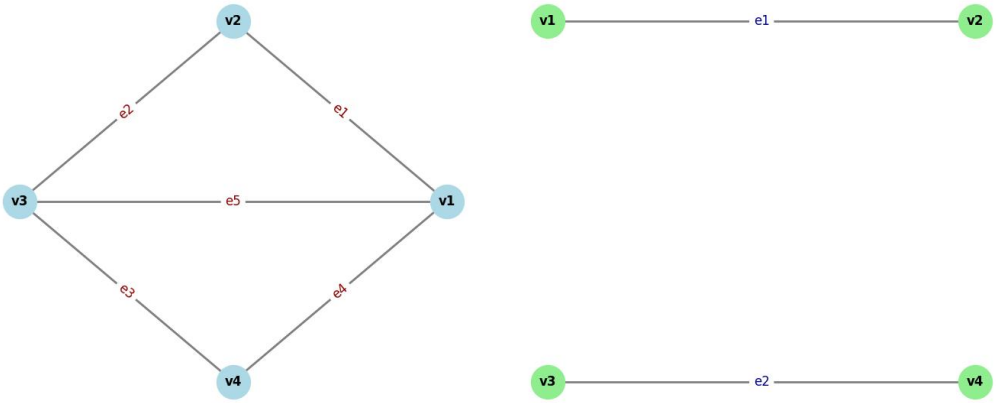
- **Grafo Conexo:** Um grafo é considerado conexo se houver um caminho entre qualquer par de vértices.
- **Grafo Desconexo:** Um grafo que não possui caminhos entre todos os pares de vértices.

Na Figura 3, apresentam-se dois grafos definidos sobre o mesmo conjunto de vértices $V = \{v_1, v_2, v_3, v_4\}$: à esquerda, um grafo conexo com cinco arestas; à direita, um grafo desconexo.

2.3.1.4 *Grafos Ponderados*

Grafo Ponderado: Um grafo onde cada aresta tem um peso ou custo associado. É frequentemente usado em problemas de otimização.

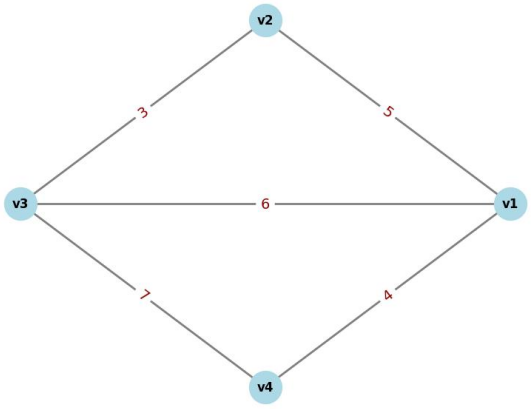
Figura 3 – Exemplo de grafo conexo (à esquerda) e de grafo desconexo (à direita).



Fonte: Próprio autor

Na Figura 4 apresenta-se um grafo ponderado definido sobre o conjunto de vértices $V = \{v_1, v_2, v_3, v_4\}$ e o conjunto de arestas ponderadas $E = \{e_1 : v_3 - v_2\,(3), e_2 : v_2 - v_1\,(5), e_3 : v_3 - v_1\,(6), e_4 : v_3 - v_4\,(7), e_5 : v_4 - v_1\,(4)\}$, em que cada aresta recebe seu peso indicado entre parênteses.

Figura 4 – Exemplo de grafo ponderado.



Fonte: Próprio autor

2.3.1.5 Subgrafos

Subgrafo: Seja $G = (V, E)$ um grafo qualquer. Um *subgrafo* $H = (V_H, E_H)$ de G é definido de forma que:

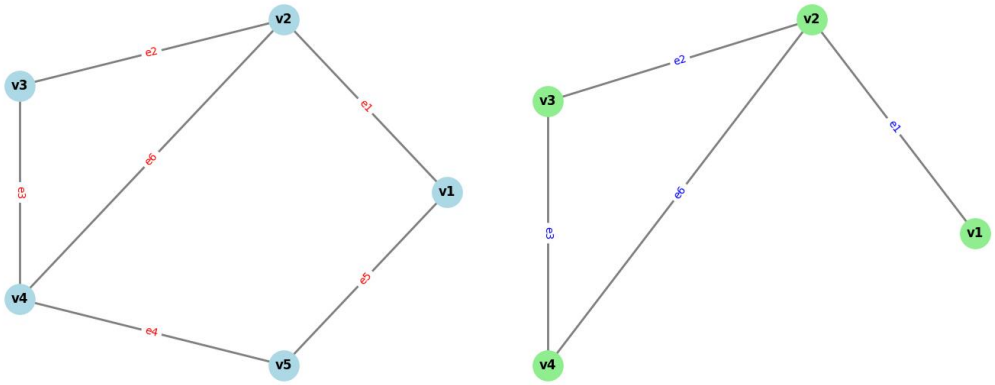
- $V_H \subseteq V$;
- $E_H \subseteq E$;

– toda aresta em E_H incide apenas em vértices de V_H, ou seja,

$$E_H \subseteq \big\{\{u,v\} \in E : u,v \in V_H\big\}.$$

Em outras palavras, retira-se de G certos vértices e/ou arestas, mantendo as incidências originais. Se, além disso, E_H contiver todas as arestas de G cujos extremos pertencem a V_H, então H é chamado de *subgrafo induzido* por V_H. Na Figura 5 apresenta-se um grafo G definido sobre o conjunto de vértices $V = \{v_1, v_2, v_3, v_4, v_5\}$ e o conjunto de arestas $E = \{e_1 : v_1 - v_2, e_2 : v_2 - v_3, e_3 : v_5 - v_1, e_4 : v_4 - v_5, e_5 : v_2 - v_4, e_6 : v_3 - v_4\}$; à direita exibe-se o subgrafo correspondente $H = (V', E')$ com $V' = \{v_1, v_2, v_3, v_4\}$ e $E' = \{e_1 : v_1 - v_2, e_2 : v_2 - v_3, e_5 : v_2 - v_4, e_6 : v_3 - v_4\}$.

Figura 5 – Exemplo de grafo (à esquerda) e um subgrafo correspondente (direita).



Fonte: Próprio autor

2.3.2 Grafos em redes sociais

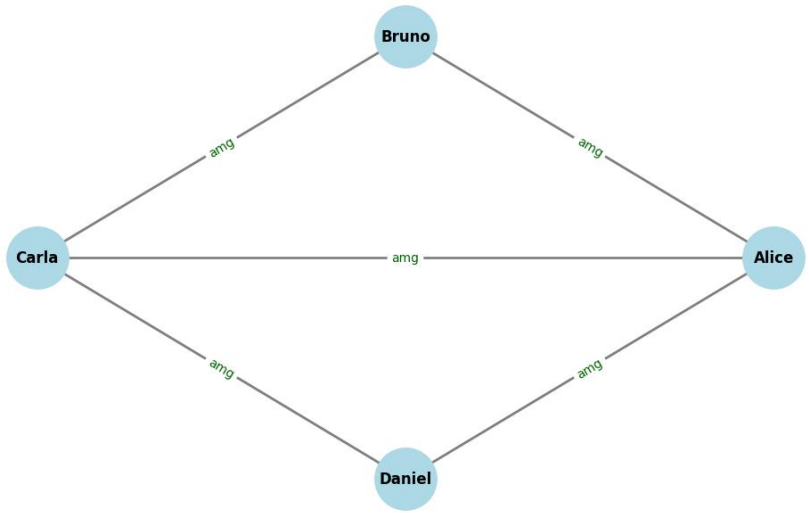
A representação de uma rede social por meio de um grafo pode ser formalizada como uma estrutura na qual os vértices simbolizam os indivíduos ou demais entidades que compõem o sistema, ao passo que as arestas evidenciam as relações ou interações estabelecidas entre esses atores. De acordo com Easley *et al.* (2010), modelagens que utilizam grafos permitem a análise de diversos aspectos das redes sociais, como a centralidade dos indivíduos, a formação de comunidades e a propagação de informações, facilitando a compreensão das dinâmicas sociais e comportamentais dentro da rede.

Um grafo que representa uma rede social como o *WhatsApp* pode funcionar de maneira que os vértices correspondem aos usuários da plataforma, enquanto as arestas representam as interações diretas entre esses usuários, como o envio de mensagens. Em um grafo

não direcionado, uma aresta entre dois vértices pode indicar uma troca mútua de mensagens entre os dois usuários. Em um grafo direcionado, a direção da aresta pode indicar quem iniciou a comunicação.

Na Figura 6, é apresentado um grafo não orientado que ilustra interações mútuas em uma rede social. Esse grafo é definido pelo conjunto de vértices (V) e pelo conjunto de arestas (E), em que cada vértice representa um usuário e cada aresta indica um vínculo de amizade (amg) entre os dois nós conectados. Na Figura 7, é apresentado um grafo direcionado definido sobre o mesmo conjunto de vértices (V) e o conjunto de arestas direcionadas (A), no qual cada aresta registra o envio de uma mensagem (msg) de um usuário para outro.

Figura 6 – Exemplo de grafo representando interações mútuas em uma rede social.

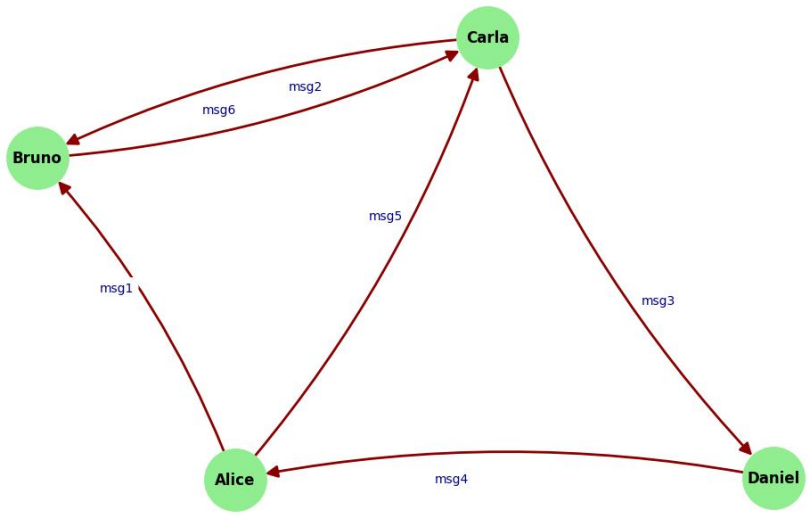


Fonte: Próprio autor

2.4 Aprendizado de máquina

Segundo Bishop e Nasrabadi (2006), o aprendizado de máquina constitui um subcampo da Ciência da Computação dedicado ao desenvolvimento de algoritmos e modelos que permitem a computadores extrair conhecimento a partir de dados e, com base nisso, realizar previsões ou tomar decisões de forma autônoma, sem depender de regras programadas explicitamente. O cerne dessa disciplina reside na capacidade de generalização: a partir de um conjunto finito de exemplos observados, o modelo é habilitado a inferir a respeito de novos insumos ainda não vistos durante o processo de treinamento (RUSSELL; NORVIG, 2016). A Figura 8 apresenta uma representação hierárquica dos principais campos relacionados à IA, por meio de círculos que ilustram o relacionamento entre esses domínios. No nível mais externo, encontra-se o campo

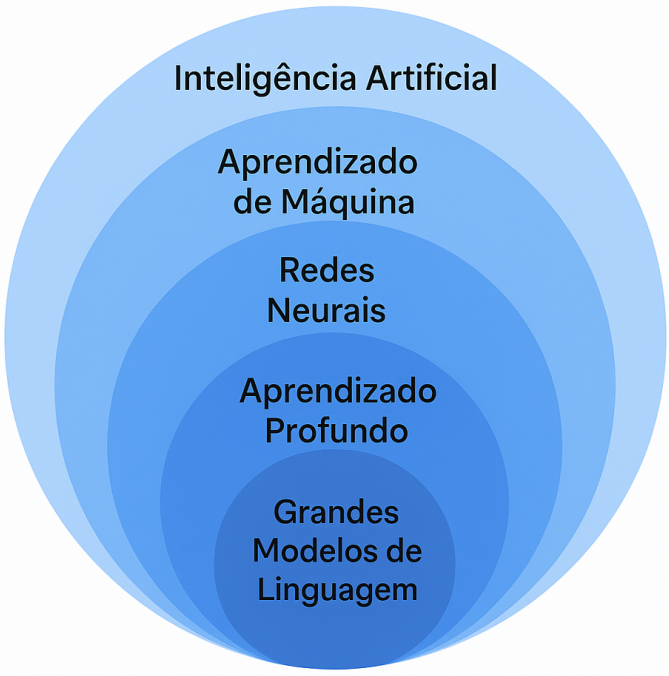
Figura 7 – Exemplo de grafo representando interações não mútuas em uma rede social.



Fonte: Próprio autor

amplo da Inteligência Artificial, que abrange todos os métodos e abordagens voltados para a criação de sistemas capazes de simular comportamentos inteligentes.

Figura 8 – Representação hierárquica dos principais campos relacionados à IA.



Fonte: Próprio autor

Os métodos de aprendizado de máquina podem ser classificados em duas categorias principais: aprendizado supervisionado e aprendizado não supervisionado. Tais abordagens distinguem-se sobretudo pelo tratamento conferido aos dados de entrada e pela presença ou não

de rótulos associados às amostras. Conforme detalhado por Bishop e Nasrabadi (2006), tais categorias podem ser formalmente caracterizadas da seguinte maneira.

2.4.1 Métodos Supervisionados

Os métodos de aprendizado de máquina supervisionado caracterizam-se pela utilização de bases de dados rotuladas, nas quais cada vetor de atributos de entrada encontra-se associado a um valor de saída previamente conhecido. O objetivo central dessa estratégia consiste em estimar, a partir dos exemplos observados no conjunto de treinamento, uma função de mapeamento capaz de generalizar e predizer com acurácia as respostas correspondentes a novas instâncias não vistas. Tal abordagem revela-se essencial em tarefas de classificação e regressão, pois permite ao modelo identificar e formalizar padrões intrínsecos aos dados, aplicando em seguida esse conhecimento para gerar estimativas confiáveis sobre observações futuras.

2.4.1.1 Classificação

A classificação é uma tarefa de aprendizado supervisionado em que, a partir de um conjunto de dados rotulado, o algoritmo aprende uma função de decisão capaz de dividir o espaço de características em regiões correspondentes a cada classe. Após o treinamento, o modelo deve generalizar bem, atribuindo corretamente rótulos a novas amostras não vistas durante o ajuste.

2.4.1.2 Regressão

A tarefa de regressão dedica-se à estimação de variáveis numéricas contínuas a partir de um conjunto de atributos de entrada. Em contraste com a classificação, cuja saída assume categorias discretas, a regressão procura modelar relações funcionais que expliquem o comportamento de uma grandeza sobre um domínio contínuo.

2.4.2 Métodos Não Supervisionados

Os métodos de aprendizagem não supervisionada constituem um arcabouço de técnicas aplicadas sobre conjuntos de dados desprovidos de rótulos de saída, de modo que as respostas corretas não se encontram previamente especificadas. Seu propósito central é revelar a estrutura latente e os padrões inerentes à distribuição dos dados, sem a interferência de uma supervisão explícita. Dentre as abordagens mais recorrentes destacam-se aquelas voltadas à

clusterização, cuja finalidade é agrupar instâncias semelhantes em compartimentos internamente homogêneos, e os métodos de redução de dimensionalidade, que buscam projetar o espaço de características em dimensões inferiores ao passo que preservam a informação e a variabilidade essenciais. Essas técnicas encontram aplicação em tarefas de exploração e visualização de dados, descoberta de conhecimento e pré-processamento para modelos subsequentes, especialmente em cenários nos quais não há rótulos de classe ou valores-alvo disponíveis.

2.5 Redes Neurais Artificiais

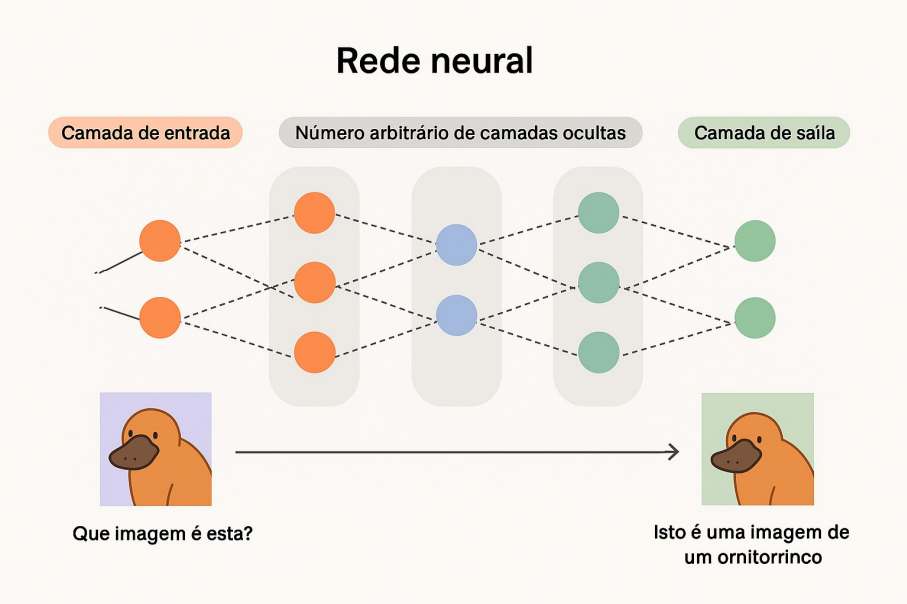
Esta seção tem por escopo a exposição dos fundamentos teóricos essenciais ao estudo das redes neurais artificiais. Os conceitos aqui desenvolvidos baseiam-se primordialmente nas obras de referência de Haykin (1994), Bishop e Nasrabadi (2006) e Goodfellow (2016), as quais apresentam de forma sistemática os aspectos teóricos e metodológicos subjacentes ao processo de treinamento de arquiteturas neurais artificiais.

As redes neurais artificiais são modelos computacionais inspirados na estrutura e no funcionamento do cérebro humano, amplamente empregadas em técnicas de aprendizado profundo para a resolução de problemas complexos. Sua arquitetura, composta por camadas interconectadas de neurônios artificiais, permite a extração de representações abstratas dos dados e a generalização de padrões, tornando-as particularmente eficazes em tarefas como classificação, regressão e reconhecimento de padrões. A Figura 9 ilustra, de forma simplificada, o funcionamento de uma rede neural artificial aplicada ao reconhecimento de imagens. O processo é representado por camadas interconectadas que recebem, processam e interpretam informações visuais com o objetivo de realizar uma classificação.

2.5.1 Neurônios

Um neurônio em uma rede neural artificial é a unidade fundamental de processamento que simula o comportamento dos neurônios biológicos. Cada neurônio recebe entradas de múltiplas fontes, realiza uma combinação linear dessas entradas, ponderadas por coeficientes denominados pesos, e aplica um termo de viés (*bias*) para ajustar o valor final da combinação.

Figura 9 – Rede neural artificial aplicada ao reconhecimento de imagens.



Fonte: Próprio autor

2.5.1.1 Pesos e Soma Ponderada

Os pesos são parâmetros que determinam a importância relativa de cada entrada no processo de tomada de decisão. Estes valores numéricos são aprendidos durante o treinamento através de algoritmos de otimização como o gradiente descendente, que ajustam iterativamente os pesos para minimizar uma função de custo. A magnitude e o sinal dos pesos codificam tanto a força quanto a natureza da relação entre as variáveis de entrada e a resposta do sistema.

A soma ponderada constitui a operação fundamental que combina linearmente as entradas utilizando os pesos como coeficientes multiplicativos. Esta operação forma a base computacional dos neurônios artificiais antes da aplicação de funções de ativação não-lineares, permitindo que o modelo capture relações complexas entre variáveis. A soma ponderada é calculada pela seguinte fórmula:

$$z = \sum_{i=1}^{n} w_i x_i + b \tag{2.2}$$

Onde:

- w_i são os *pesos* atribuídos a cada entrada x_i,
- x_i representam as *entradas*,
- b é o *termo de viés* (*bias*), que ajusta a soma ponderada de maneira a melhorar a flexibilidade do modelo.

Esse cálculo da soma ponderada, ou combinação linear das entradas, permite o funcionamento do neurônio, fornecendo o valor z que será processado na próxima etapa.

2.5.1.2 Função de Ativação

As funções de ativação desempenham um papel fundamental na introdução de não-linearidades nos modelos, permitindo que as redes neurais adquiram a capacidade de aproximação universal. Essas funções matemáticas são aplicadas à soma ponderada das entradas de cada neurônio, definindo se, e com qual intensidade, o neurônio deve ser ativado e transmitir seu sinal à camada seguinte. Dentre as funções mais utilizadas, destaca-se a Rectified Linear Unit (ReLU).

A função ReLU é definida matematicamente como:

$$f(x) = \begin{cases} x, & \text{se } x > 0 \\ 0, & \text{caso contrário} \end{cases} \tag{2.3}$$

Onde:

- $f(x)$ é a saída da função de ativação,
- x representa a entrada do neurônio (isto é, a soma ponderada dos sinais recebidos).

Essa formulação simples retorna o próprio valor de entrada quando este é positivo, e zero nos casos em que o valor é negativo ou nulo. Sua ampla adoção em redes neurais profundas deve-se à sua simplicidade computacional e à capacidade de acelerar a convergência durante o treinamento.

2.5.2 Camadas

As camadas constituem unidades organizacionais que estabelecem uma arquitetura hierárquica que governa o processamento sequencial da informação através da rede. Cada camada é formada por um conjunto de neurônios artificiais interconectados que executam transformações matemáticas específicas nos dados de entrada, aplicando operações de soma ponderada seguidas de funções de ativação não-lineares.

A estruturação em camadas permite a decomposição de problemas complexos em representações intermediárias progressivamente mais abstratas, onde cada nível hierárquico extrai características de diferentes níveis de complexidade dos dados originais. As redes neurais

artificiais são estruturadas através de uma arquitetura estratificada que compreende três categorias fundamentais de camadas:

- **Camada de Entrada**: Constitui o ponto inicial da arquitetura neural, sendo responsável pela recepção e preprocessamento dos dados provenientes do conjunto de dados. Nesta etapa, não são realizadas transformações computacionais complexas; sua função primordial consiste na transmissão fidedigna dos valores de entrada para as camadas subsequentes, preservando a integridade dimensional dos dados originais.
- **Camadas Ocultas**: Situadas no estrato intermediário da arquitetura neural, representam o núcleo computacional da rede, onde ocorre o processamento mais sofisticado dos dados. Cada unidade neuronal executa uma transformação matemática que envolve o cálculo de uma combinação linear ponderada dos sinais de entrada da camada precedente, seguida da aplicação de uma função de ativação não linear. A utilização de múltiplas camadas ocultas em cascata caracteriza as arquiteturas de aprendizado profundo, conferindo à rede a capacidade de extrair e codificar padrões hierárquicos de crescente complexidade.
- **Camada de Saída**: Corresponde ao estágio terminal da arquitetura neural, sendo responsável pela geração da resposta final do modelo, a qual se manifesta sob a forma de predições quantitativas ou classificações categóricas.

2.5.3 Processo de Treinamento

O processo de treinamento em redes neurais artificiais compreende três etapas fundamentais: a propagação direta (*feedforward*), a computação da função de perda e a retropropagação (*Backpropagation*):

2.5.3.1 Propagação direta (feedforward)

Neste processo, os dados de entrada são propagados através da arquitetura da rede, sequencialmente por cada camada, até a obtenção da saída final. Esta propagação constitui etapa fundamental, uma vez que possibilita à rede computar a saída baseada nos parâmetros de peso atuais, precedendo quaisquer ajustes nos mesmos.

2.5.3.2 Função de Perda

Subsequentemente à geração da saída, a rede neural necessita comparar o valor predito com o valor esperado, empregando uma *função de perda*. A função de perda quantifica a discrepância entre a predição da rede e o valor de referência. Tal função fornece uma métrica quantitativa acerca da qualidade preditiva da rede, estabelecendo um critério objetivo para avaliação do desempenho do modelo.

2.5.3.3 Retropropagação (backpropagation)

Este processo fundamenta-se na computação dos *gradientes* da função de perda em relação a cada parâmetro de peso da rede, possibilitando a identificação da direção na qual os pesos devem ser modificados para minimizar o erro do modelo. O algoritmo mais comum para ajustar os pesos durante a retropropagação é o *gradient descent*. A equação de atualização de peso pode ser expressa como:

$$w_{novo} = w_{atual} - \eta \cdot \frac{\partial L}{\partial w} \tag{2.4}$$

Onde:

- w_{novo} é o valor atualizado do peso,
- w_{atual} é o valor atual do peso,
- η é a *taxa de aprendizado* (*learning rate*), que controla o tamanho do ajuste,
- $\frac{\partial L}{\partial w}$ é o gradiente da função de perda L em relação ao peso w.

A retropropagação constitui o núcleo do processo de aprendizado em redes neurais, uma vez que viabiliza a propagação dos erros desde a camada de saída até as camadas ocultas e de entrada, ajustando os parâmetros de forma a otimizar a função de perda. Este processo iterativo é executado até que a rede atinja um nível de erro tolerável ou que o critério de convergência seja satisfeito.

2.5.4 Regularização

A regularização constitui um conjunto de técnicas concebidas para prevenir que o modelo assimile padrões espúrios ou características irrelevantes presentes nos dados de treinamento, fenômeno denominado *overfitting*. O *overfitting* manifesta-se quando o modelo se ajusta

excessivamente aos dados de treinamento, comprometendo sua capacidade de generalização na predição de novos dados. As técnicas de regularização mais extensivamente empregadas em redes neurais são *dropout*, parada antecipada (*early stopping*) e *regularização L2*.

2.5.4.1 Dropout

O *dropout* é uma técnica utilizada em redes neurais que consiste na desativação aleatória de uma fração de neurônios durante o treinamento, de acordo com uma probabilidade pré-definida para cada neurônio. Esse processo força o modelo a aprender representações mais robustas e distribuídas, evitando a dependência excessiva de subconjuntos específicos de neurônios e reduzindo o risco de co-adaptações, o que contribui para mitigar o *overfitting*, especialmente em redes profundas.

2.5.4.2 Parada antecipada (early stopping)

A parada antecipada opera mediante o monitoramento do desempenho em um conjunto de validação durante o processo de aprendizagem, interrompendo o treinamento quando tal desempenho cessa de apresentar melhorias consecutivas. Tal comportamento indica que o modelo pode estar iniciando um processo de memorização dos dados de treinamento, comprometendo sua capacidade de generalização para dados não observados anteriormente.

2.5.4.3 Regularização L2

Trata-se de uma técnica que adiciona um termo de penalização à função de perda da rede neural, incentivando a manutenção de pesos pequenos e evitando que os parâmetros assumam valores elevados apenas para minimizar o erro de treinamento. Tecnicamente, isto implica que, durante o processo de aprendizado, a cada atualização dos pesos, aplica-se uma componente de decaimento que os direciona em direção ao valor zero. Este procedimento, denominado também *weight decay*, impede que o modelo desenvolva complexidade excessiva ou sensibilidade pronunciada a ruídos presentes nos dados. Matematicamente, a regularização L2 pode ser expresso como:

$$L(w) = L_0(w) + \lambda \sum_i w_i^2 \tag{2.5}$$

Onde:

- $L(w)$ é a nova função de perda regularizada,
- $L_0(w)$ é a função de perda original,
- λ é o hiperparâmetro que controla a força da regularização,
- w_i são os pesos da rede.

O termo adicional, $\lambda \sum_i w_i^2$, impõe um custo por ter pesos grandes. O valor de λ regula o impacto da regularização no modelo: valores maiores de λ forçam os pesos a serem menores, enquanto valores menores permitem que os pesos cresçam mais livremente. O resultado dessa penalização é um modelo com pesos menores e, portanto, *menos complexo*, o que tende a generalizar melhor quando confrontado com dados desconhecidos.

2.5.5 Hiperparâmetros

Constituem variáveis externas ao modelo que regulam o comportamento do processo de treinamento em redes neurais artificiais. Em contraposição aos parâmetros da rede, tais como pesos e vieses, que são otimizados durante o treinamento, os hiperparâmetros necessitam ser estabelecidos previamente ao início do processo de treinamento e exercem influência substancial sobre o desempenho final do modelo. A seleção apropriada dos hiperparâmetros pode aprimorar tanto a velocidade de convergência quanto a capacidade de generalização da rede. Subsequentemente, apresentam-se alguns dos principais hiperparâmetros.

2.5.5.1 Taxa de Aprendizado (Learning Rate)

Determina a magnitude dos incrementos que o algoritmo de otimização executa ao ajustar os parâmetros da rede com base nos gradientes da função de perda. Um valor apropriado da taxa de aprendizado é essencial para assegurar uma convergência eficiente do modelo. Caso a taxa de aprendizado seja excessivamente elevada, o modelo pode ultrapassar o ponto ótimo, resultando em oscilações ou mesmo em falha de convergência. Inversamente, se a taxa de aprendizado for demasiadamente reduzida, o modelo pode convergir de forma excessivamente lenta, demandando um número elevado de iterações para atingir um ponto satisfatório.

2.5.5.2 Número de Camadas e Neurônios

O número de camadas e neurônios em uma rede neural define sua profundidade e largura, impactando diretamente sua capacidade de aprender e representar padrões complexos

dos dados. Redes mais profundas podem capturar abstrações hierárquicas, mas são suscetíveis a problemas como desvanecimento do gradiente e *overfitting* se não houver regularização adequada. Já o número de neurônios por camada afeta a capacidade de captar variações nos dados, sendo que um excesso pode aumentar o risco de *overfitting*.

2.5.5.3 Número de Épocas (Epochs)

Refere-se ao número de iterações durante as quais o conjunto de dados completo é processado pela rede ao longo do processo de treinamento. Cada época corresponde a uma passagem completa sobre a totalidade dos exemplos de treinamento, possibilitando que o modelo ajuste seus parâmetros com base em cada amostra de dados.

Caso o número de épocas seja insuficiente, a rede pode não dispor de tempo adequado para assimilar apropriadamente os padrões presentes nos dados, resultando em um modelo subajustado (*underfitting*). Contudo, se o número de épocas for excessivo, o modelo pode desenvolver sobreajuste aos dados de treinamento, culminando em *overfitting*.

2.5.6 Avaliação dos modelos

Para mensurar o desempenho dos modelos, são calculadas as métricas mais comumente utilizadas em tarefas de classificação binária. A seguir, apresenta-se uma breve descrição de cada métrica e seu funcionamento:

– **Acurácia** (accuracy): define-se como a razão entre o número de exemplos corretamente classificados (verdadeiros positivos *TP* mais verdadeiros negativos *TN*) e o total de exemplos avaliados (*P*ositivos mais *N*egativos). Formalmente,

$$accuracy = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}} \tag{2.6}$$

onde FP e FN correspondem, respectivamente, aos falsos positivos e falsos negativos.

– **Precisão** (precision): trata-se de uma métrica que avalia a proporção de exemplos classificados como positivos que realmente pertencem à classe positiva. Calcula-se por:

$$precision = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{2.7}$$

Valores altos de precisão indicam que poucos exemplos negativos foram incorretamente classificados como positivos.

– **Recall** quantifica a proporção de exemplos positivos que foram corretamente identificados pelo modelo:

$$recall \;=\; \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{2.8}$$

Um alto recall significa que o modelo cometeu poucos erros do tipo falso negativo (FN).

– $\mathbf{F}_1$ (f1): corresponde à média harmônica entre precisão e recall, fornecendo um único valor que busca balancear esses dois aspectos:

$$\mathrm{f1} \;=\; 2 \times \frac{\mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} \tag{2.9}$$

O F_1 só será alto quando tanto precisão quanto recall apresentarem desempenho satisfatório.

– **Área sob a curva ROC** (roc_auc_score): sempre que há possibilidade de atribuir ao modelo uma pontuação contínua (como probabilidades), constrói-se a *Receiver Operating Characteristic* (ROC), que relaciona:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \quad \text{e} \quad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}\,. \tag{2.10}$$

A roc_auc_score mede a área sob essa curva, variando de 0,0 a 1,0. Valores mais próximos de 1,0 indicam melhor desempenho.

Em síntese, a combinação dessas métricas permite entender não apenas o percentual geral de acertos, mas também a qualidade das previsões positivas e a capacidade de capturar instâncias críticas.

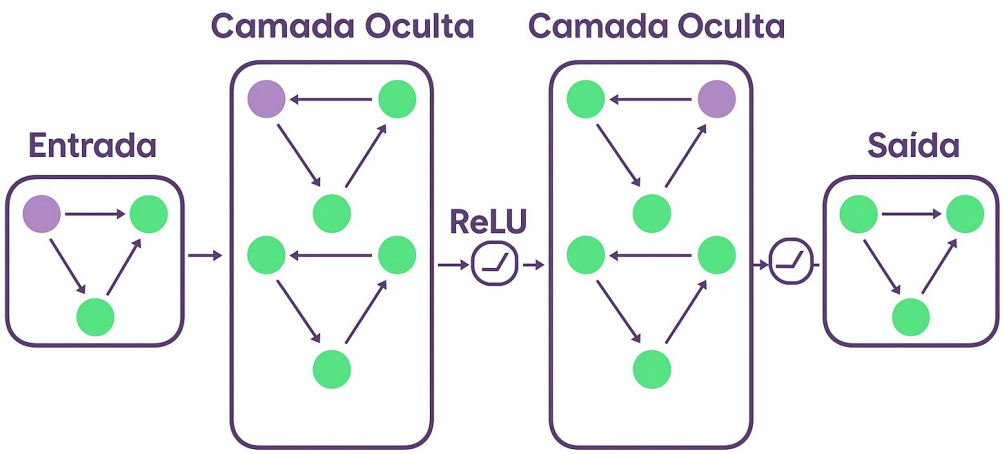
2.6 Redes Neurais para Grafos

As redes neurais para grafos podem ser caracterizadas como um modelo de aprendizado profundo que estende as capacidades das redes neurais convencionais, de modo a viabilizar sua aplicação direta sobre dados estruturados na forma de grafos (HAMILTON, 2020). Distintamente das redes neurais tradicionais, que processam os dados de forma independente ou mediante estruturas sequenciais, as GNN são concebidas especificamente para explorar a estrutura topológica dos grafos. Este processo ocorre mediante a captura das dependências existentes entre os vértices, os quais se encontram intrinsecamente interconectados pelas arestas que constituem o grafo (ZHOU *et al.*, 2020).

Khemani *et al.* (2024) destaca que as GNN são capazes de processar tanto as características intrínsecas de cada nó individual, quanto a informação derivada de suas conexões com

outros nós no grafo. A Figura 10 apresenta a arquitetura de uma rede neural para grafos composta de quatro componentes fundamentais: uma camada de entrada responsável pela codificação inicial dos dados, duas camadas ocultas intermediárias que processam as representações latentes dos nós, intercaladas por uma função de ativação ReLU, e uma camada de saída que produz as predições finais.

Figura 10 – Exemplo simples de arquitetura de uma rede neural para grafos



Fonte: Próprio autor

2.6.1 Funcionamento de Redes Neurais para Grafos

O funcionamento de uma *Graph Neural Network* pode ser explicado em termos de um processo iterativo de aprendizado e atualização das representações dos nós, fundamentado nas seguintes fases.

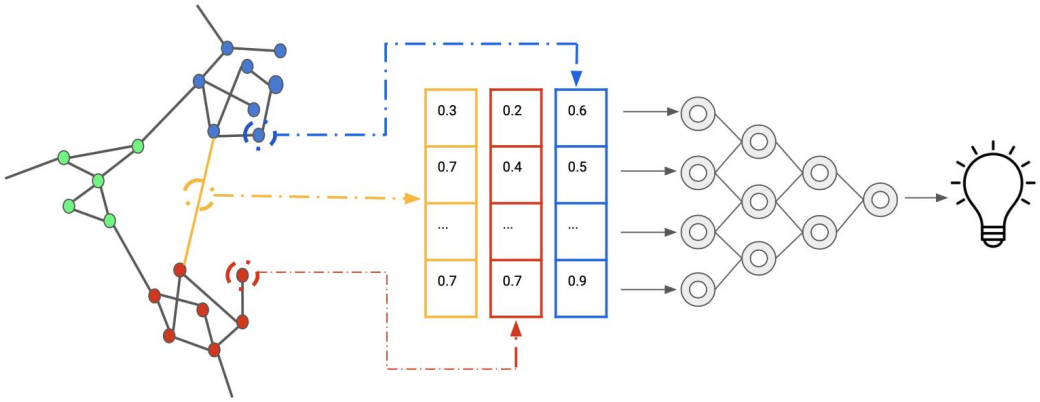
2.6.1.1 Inicialização das Representações dos Nós

Inicialmente, cada nó de um grafo é associado a um vetor de características, também denominado *embedding*. Tal como proposto por Mikolov *et al.* (2013), *embedding* constitui uma técnica que representa objetos, como nós de um grafo, em um espaço vetorial contínuo de baixa dimensionalidade, facilitando a captura de suas características semânticas e estruturais de forma compacta e eficiente. Os *embedding* transformam dados discretos em vetores densos,

possibilitando que similaridades e relações sejam expressas mediante proximidade geométrica no espaço vetorial.

A Figura 11 ilustra o fluxo de processamento de dados em um sistema de aprendizado de máquina baseado em grafos. Inicialmente, temos um grafo de conhecimento, onde cada nó representa uma entidade e as conexões representam relações entre elas. Em seguida, esse grafo é convertido em representações vetoriais (*embeddings*), que codificam as características estruturais e relacionais dos nós. Essas representações embutidas são então utilizadas como entrada para um modelo de aprendizado de máquina, como uma rede neural, que processa os dados e realiza a tarefa desejada.

Figura 11 – Como um grafo é convertido em *embeddings* vetoriais, que resumem suas informações estruturais para serem usados em tarefas de aprendizado de máquina.



Fonte: SILVA, João (2019), Wikimedia Commons

2.6.1.2 Agregação de Vizinhança

A principal inovação das GNN reside na capacidade de agregar informações de vizinhança, ou seja, combinar as características de um nó com as características de seus nós vizinhos, respeitando a estrutura topológica do grafo. Conforme estabelecido por Kipf e Welling (2016), o processo de agregação de vizinhança ocorre de maneira iterativa, em camadas sucessivas da rede neural, nas quais cada nó atualiza sua representação com base em uma função de agregação aplicada às características dos nós a ele conectados.

Em cada camada k de uma GNN, a atualização do vetor de características de um nó v é realizada por meio de uma função de agregação $\mathrm{AGG}^{(k)}$. Essa função combina a representação anterior do próprio nó com as representações anteriores de todos os seus vizinhos (CORSO *et al.*, 2020). Formalmente:

$$\mathbf{h}_v^{(k)} \;=\; \mathrm{AGG}^{(k)}\Big(\mathbf{h}_v^{(k-1)},\; \big\{\mathbf{h}_u^{(k-1)} : u \in \mathscr{N}(v)\big\}\Big), \tag{2.11}$$

Onde:

- $\mathbf{h}_v^{(k-1)}$ é o vetor de características (*embedding*) do nó v na camada $(k-1)$.
- $\mathscr{N}(v)$ denota o conjunto de vizinhos de v no grafo.
- $\{\mathbf{h}_u^{(k-1)} : u \in \mathscr{N}(v)\}$ é o multiconjunto dos vetores de características dos nós vizinhos de v na camada $(k-1)$.

Em outras palavras, a cada camada, cada nó coleta informação de seus próprios atributos e dos atributos de seus vizinhos, combinando esses valores por meio de $\mathrm{AGG}^{(k)}$ para gerar sua nova representação $\mathbf{h}_v^{(k)}$. Dessa forma, a rede é capaz de propagar informação local ao longo das camadas, incorporando progressivamente o contexto estrutural do grafo (CORSO *et al.*, 2020).

As funções de agregação podem variar a depender da arquitetura da GNN, podendo constituir uma média ponderada, uma soma, ou mecanismos mais sofisticados, tais como o mecanismo de atenção, que possibilita ao modelo atribuir diferentes pesos aos vizinhos, em função de sua relevância.

Após a fase de agregação, cada nó emprega uma função de atualização para modificar suas características com base nas informações coletadas de seus vizinhos (HAMILTON *et al.*, 2017). Esta função de atualização geralmente consiste em duas etapas principais: primeiramente, realiza-se uma combinação linear das características agregadas, normalmente por meio de uma multiplicação por uma matriz de pesos aprendida durante o treinamento, seguida de uma função de ativação não linear, como a ReLU. O objetivo desta etapa consiste em permitir que o nó atualize sua representação de forma a refletir tanto suas características iniciais quanto a informação recebida de sua vizinhança, refinando progressivamente sua representação ao longo das camadas da rede (HAMILTON *et al.*, 2017).

2.6.1.3 Propagação em Múltiplas Camadas

O processo de agregação e atualização é iterado ao longo de múltiplas camadas da GNN. A cada camada, a informação é propagada para distâncias progressivamente maiores no grafo, o que possibilita que um nó não apenas agregue informações de seus vizinhos imediatos, mas também capture as características de nós mais distantes, mediante sucessivas iterações

(ZHOU *et al.*, 2022).

Nas redes neurais tradicionais, cada camada é composta por um conjunto distinto de neurônios responsáveis por aprender diferentes representações dos dados de entrada. Em contraste, em uma GNN, o conceito de camada está intrinsecamente ligado ao processo de propagação de mensagens no grafo, em que os mesmos nós atualizam iterativamente seus estados ao agregarem informações de seus vizinhos em múltiplas rodadas de comunicação, mantendo a estrutura do grafo inalterada (WU *et al.*, 2020).

2.6.1.4 Predição

Após múltiplas camadas de agregação e atualização, as representações finais de cada nó capturam informações relevantes não apenas de suas características iniciais, mas também da estrutura local e global do grafo. Conforme Kipf e Welling (2016), tais representações finais podem ser empregadas para diversas tarefas de predição, tais como classificação de nós (por exemplo, predizer a categoria ou rótulo de um nó), classificação de arestas (predizer a existência ou tipo de conexão entre dois nós) ou predição de propriedades globais do grafo (tal como a predição de uma propriedade em nível de grafo, empregada em análise molecular ou de redes sociais).

2.6.2 Arquiteturas GNN

Ao longo dos últimos anos, diversas arquiteturas de GNN foram propostas, cada uma visando solucionar desafios específicos e otimizar o desempenho em diferentes cenários. A seguir, serão apresentadas as arquiteturas mais relevantes para o presente trabalho, detalhando-se suas características, vantagens e desvantagens.

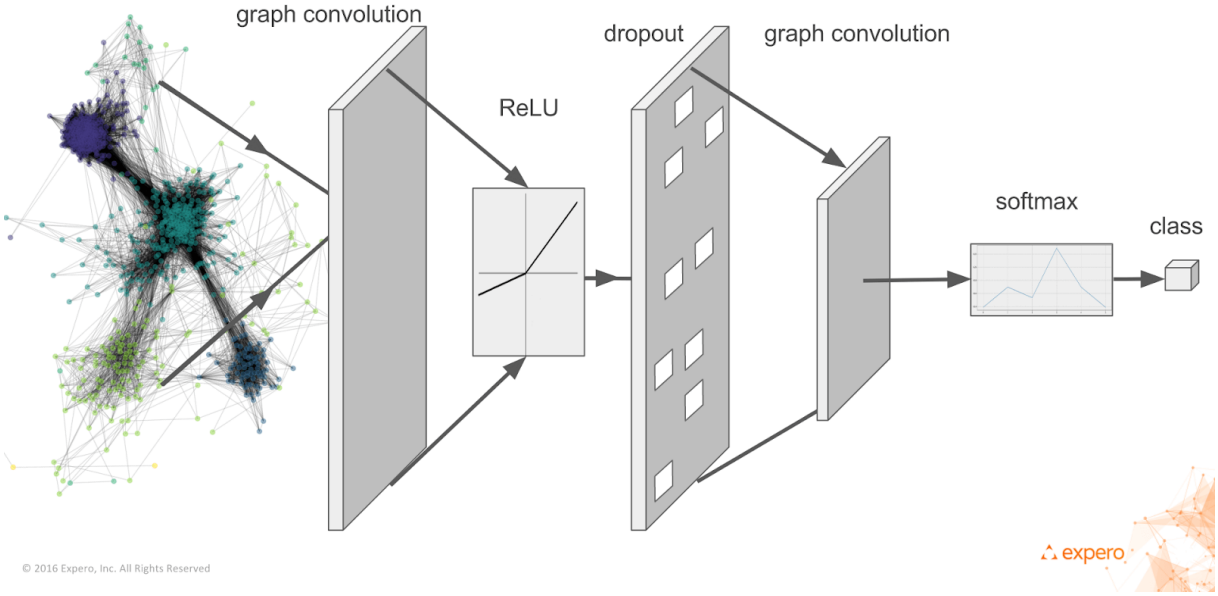
2.6.2.1 Graph Convolutional Networks (GCN)

O modelo GCN, conforme proposto por Kipf e Welling (2016), é uma das arquiteturas pioneiras e influentes que adaptou a convolução de domínios regulares (como imagens) para grafos, lidando com o desafio de processar estruturas onde a vizinhança de cada nó é de tamanho variável e sem ordenação, faltam coordenadas para guiar a agregação, e a propagação de informação precisa ser regulada para capturar dependências de múltiplos saltos sem causar excesso de suavização das características.

O mecanismo central do GCN alicerça-se em uma operação de filtragem espectral localizada, que pode ser conceituada como uma modalidade de média ponderada das características nodais e de seus adjacentes. A cada camada da arquitetura, a representação de um nó é refinada através da integração de suas propriedades intrínsecas com aquelas de seus vizinhos, empregando uma normalização fundamentada nos graus nodais (KIPF; WELLING, 2016).

A Figura 12 ilustra o funcionamento de uma Rede Neural Convolucional em Grafos GCN. O processo inicia com a entrada de um grafo, cujos nós e conexões são processados por camadas de convolução em grafos, responsáveis por agregar informações dos vizinhos. Em seguida, aplica-se uma função de ativação ReLU e um *dropout* para evitar *overfitting*. Após uma nova convolução, os dados passam por uma função *softmax*, que gera a classificação final dos nós.

Figura 12 – Exemplo de Rede Neural Convolucional em Grafos



Fonte: Ganssle (2018)

Não obstante sua relevância histórica e eficácia comprovada em diversas aplicações, o GCN apresenta limitações intrínsecas que motivaram consideráveis avanços subsequentes no domínio das GNN. Conforme evidenciado por Khemani *et al.* (2024), a formulação original do GCN confronta desafios significativos associados ao fenômeno de *oversmoothing* (*over-smoothing*), no qual a aplicação de múltiplas camadas convolucionais resulta na homogeneização excessiva das representações nodais, ocasionando potencial perda de informações discriminativas imprescindíveis.

2.6.2.2 Graph Convolutional Networks II (GCNII)

O modelo GCNII, conforme proposto por Chen *et al.* (2020), constitui um aprimoramento do GCN original ao viabilizar a edificação de arquiteturas substancialmente mais profundas sem sucumbir ao fenômeno de *oversmoothing* característico das estruturas GCN tradicionais. O *oversmoothing* é o fenômeno em que, ao empilhar muitas camadas de uma GNN, as representações dos nós convergem para vetores muito parecidos (colapso para um subespaço de baixa variância), perdendo poder discriminativo. Ao passo que o GCN convencional agrega as informações nodais com aquelas de seus adjacentes de maneira direta, o GCNII incorpora duas abordagens fundamentais para salvaguardar informações primordiais e otimizar o treinamento em profundidade: a conectividade residual inicial e o mapeamento identitário.

Segundo Chen *et al.* (2020), a conectividade residual inicial assegura que, a cada camada, uma parcela das características primordiais de entrada seja mantida e propagada posteriormente, prevenindo que o sinal se deteriore conforme a arquitetura se aprofunda. Por sua vez, o mapeamento identitário equilibra dinamicamente a aplicação da transformação adquirida pelo estrato com a mera propagação de informações inalteradas, mitigando o risco de que camadas consecutivas filtrem de forma excessiva as características pertinentes (CHEN *et al.*, 2020).

Apesar de proporcionar tais progressos, o GCNII evidencia determinadas limitações operacionais. Conforme elucidado por Chen *et al.* (2020), a demanda computacional por estrato é moderadamente superior àquela do GCN convencional, porquanto se requer a manutenção e combinação de múltiplas matrizes a cada iteração, o que pode intensificar o consumo de memória em estruturas de larga escala.

2.6.2.3 Graph Attention Networks (GAT)

Conforme descrito por Velickovic *et al.* (2017), o GAT foi desenvolvido para superar uma limitação central das GNN anteriores: a dificuldade em distinguir a importância relativa dos nós vizinhos. Para tanto, emprega mecanismos de atenção que atribuem pesos individuais às arestas de vizinhança, permitindo que a rede aprenda quais conexões exercem maior influência sobre a tarefa em questão. O mecanismo de atenção é um procedimento de aprendizado que calcula, a partir das características dos nós (e opcionalmente das arestas), pesos que quantificam a relevância de cada vizinho. Esses pesos são normalizados e usados para ponderar a agregação de informações, permitindo que o modelo concentre-se nos elementos mais informativos.

Segundo Velickovic *et al.* (2017), o mecanismo central do GAT fundamenta-se em camadas de atenção que computam coeficientes de atenção para cada par de nós conectados. Estes coeficientes determinam o peso da contribuição de cada adjacente para a atualização da representação do nó central. Esta estratégia possibilita que o GAT apreenda distintos aspectos das relações entre nós, revelando-se capaz de adaptar-se a diversificados tipos de estruturas e tarefas (VELICKOVIC *et al.*, 2017).

Embora demonstre eficácia, o GAT manifesta desafios específicos relacionados ao custo computacional elevado, particularmente em estruturas de grafo de larga escala. Wu *et al.* (2020) assinala que este custo computacional pode restringir a aplicabilidade do GAT em cenários industriais com limitações temporais e de recursos.

2.6.2.4 GraphSAGE (Graph Sample and Aggregation)

Conforme proposto por Hamilton *et al.* (2017), o GraphSAGE foi concebido para solucionar os desafios inerentes ao aprendizado em grafos de larga escala, particularmente aqueles que exibem dinâmicas temporais ou encontram-se em expansão contínua. O GraphSAGE introduz uma metodologia inovadora que viabiliza a edificação de representações para nós não observados durante o treinamento, revelando-se especialmente pertinente para cenários que englobam sistemas dinâmicos ou estruturas de grafos em transformação constante.

A principal contribuição do GraphSAGE reside na introdução de um mecanismo de amostragem e agregação de adjacentes. Em detrimento do processamento de todos os vizinhos de um nó, o modelo seleciona estocasticamente um subconjunto fixo de adjacentes, mitigando a complexidade computacional (HAMILTON *et al.*, 2017). Essa etapa pode ser formalizada da seguinte forma:

$$h_v^{(k)} = \sigma\left(W^{(k)} \cdot \text{AGGREGATE}^{(k)}\left(h_u^{(k-1)}, |, u \in \mathcal{N}_S(v) \cup h_v^{(k-1)}\right)\right) \tag{2.12}$$

onde:

- $h_v^{(k)}$ é a representação do nó v na camada k,
- $\mathcal{N}_S(v)$ é um subconjunto amostrado de vizinhos do nó v,
- $\text{AGGREGATE}^{(k)}$ é a função de agregação utilizada (por exemplo, média, soma ou LSTM),
- $W^{(k)}$ é a matriz de pesos treináveis da camada k,
- σ é uma função de ativação não linear (como a ReLU).

Em vez de usar todos os vizinhos $\mathcal{N}(v)$ do nó v, o modelo seleciona um subconjunto amostrado $\mathcal{N}_S(v) \subseteq \mathcal{N}(v)$ com tamanho fixo ou limitado. Essa restrição no tamanho de $\mathcal{N}_S(v)$ é que reduz o custo computacional, porque a agregação é feita somente sobre esse subconjunto, tornando o cálculo mais leve e escalável (HAMILTON *et al.*, 2017).

Conforme assinalado por Khemani *et al.* (2024), a estratégia de amostragem de subconjuntos de adjacentes, não obstante sua eficiência, pode ocasionar a omissão de informações pertinentes, especialmente em grafos caracterizados por conectividade heterogênea, nos quais a distribuição de arestas entre os nós é significativamente variável. Outrossim, o desempenho do modelo encontra-se substancialmente condicionado à seleção apropriada da função de agregação; escolhas inadequadas podem comprometer a capacidade do modelo de apreender as complexidades estruturais inerentes à estrutura do grafo.

2.6.2.5 Approximate Personalized Propagation of Neural Predictions (APPNP)

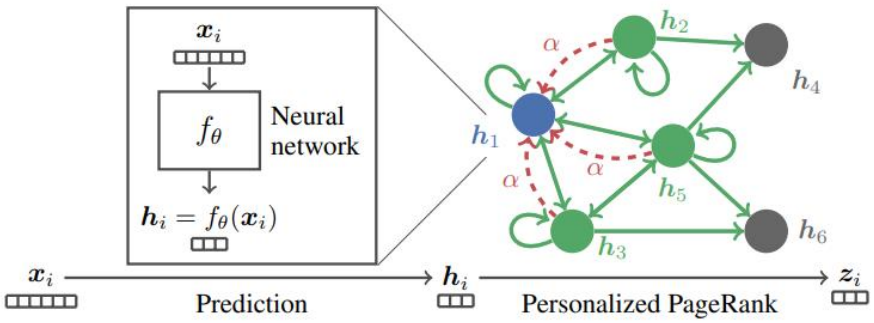
A arquitetura APPNP, conforme descrita por Gasteiger *et al.* (2018), constitui uma abordagem que amalgama as vantagens de métodos de previsão fundamentados em redes neurais com o mecanismo de propagação baseado em *pageRank* personalizado. Enquanto arquiteturas convencionais de GNN integram aprendizado e propagação em uma única etapa, o APPNP dissocia explicitamente a fase de predição, operacionalizada por uma rede neural elementar, da fase de propagação, inspirada no Personalized PageRank (PPR).

O APPNP opera fundamentalmente em duas fases distintas. Primeiramente, uma rede neural elementar realiza uma predição inicial para cada nó baseando-se em suas características intrínsecas. Subsequentemente, essas predições são refinadas de modo iterativo, difundindo parcialmente a informação ao longo das conexões da estrutura do grafo. Desta forma, a arquitetura consegue propagar informações locais pela estrutura do grafo sem comprometer integralmente o conhecimento adquirido na fase inicial Gasteiger *et al.* (2018).

A Figura 13 ilustra a propagação personalizada de previsões neurais APPNP. Inicialmente, as previsões são geradas a partir das características individuais de cada nó por meio de uma rede neural. Em seguida, essas previsões são propagadas utilizando uma adaptação do algoritmo PageRank personalizado. O modelo é treinado de forma integrada, permitindo que a rede neural e o mecanismo de propagação colaborem para melhorar os resultados.

O APPNP apresenta algumas limitações práticas que motivaram investigações subsequentes. Primeiramente, determinar quantas iterações realizar no processo de propagação e

Figura 13 – Exemplo de Rede APPNP



Fonte: Gasteiger *et al.* (2018)

quanta ênfase atribuir à predição inicial constitui um desafio metodológico: caso estes ajustes não sejam adequados, a arquitetura pode propagar informações excessivamente, tornando as representações demasiadamente homogêneas entre si, ou propagar insuficientemente, deixando de explorar a topologia do grafo (GASTEIGER *et al.*, 2018).

2.7 Conclusão

Em suma, este capítulo reuniu a fundamentação necessária para o trabalho: definimos desinformação, suas manifestações e caracterizamos os agentes de sua difusão. Apresentamos conceitos essenciais da Teoria dos Grafos para modelar interações em redes sociais e os princípios do Aprendizado de Máquina e das Redes Neurais (arquitetura, treinamento, regularização e métricas de avaliação). Por fim, introduzimos as Redes Neurais para Grafos (GNN) e suas variantes empregadas no presente trabalho, destacando seu potencial e limitações para capturar padrões topológicos relevantes à detecção de desinformadores.

3 TRABALHOS RELACIONADOS

A disseminação de desinformação em ambientes online tem crescido rapidamente, alimentada tanto por agentes mal-intencionados quanto por usuários que replicam conteúdo falso, intencionalmente ou não. Diante disso, a comunidade científica volta-se para a identificação e caracterização desses atores, buscando entender seus perfis e comportamentos nas redes sociais. Para tal, empregam-se técnicas que vão de análises comportamentais a algoritmos de aprendizado de máquina e mineração de dados em grafos. Neste capítulo, são revisadas quatro pesquisas que ilustram os principais desafios e os progressos conquistados na detecção e no combate aos impactos da desinformação.

3.1 Identifying and Characterizing Superspreaders of Low-Credibility Content on Twitter

O estudo de DeVerna *et al.* (2024) focaliza a identificação e caracterização dos denominados *superspreaders*: atores que exercem influência desproporcional na propagação de conteúdo de baixa credibilidade em plataformas de redes sociais. A investigação aborda uma problemática fundamental no enfrentamento da desinformação: a determinação dos principais vetores responsáveis pela amplificação de conteúdos enganosos em escala massiva. Os pesquisadores argumentam que a compreensão das características comportamentais e estruturais desses *superspreaders* constitui elemento fundamental para a elaboração de estratégias eficazes de contenção da desinformação no ambiente digital.

Para a investigação dos *superspreaders*, os autores desenvolveram um conjunto de métricas quantitativas que possibilitam a identificação desses atores mediante análise de seu volume de compartilhamento e grau de influência estrutural na rede social. A metodologia empregada incorporou técnicas de análise estatística direcionadas ao estabelecimento de correlações entre características individuais dos *superspreaders* e sua capacidade de exercer influência sobre outros usuários da plataforma.

Os resultados mostraram que os *superspreaders* são variados, incluindo influenciadores políticos, veículos de mídia que espalham desinformação e contas ligadas a essas fontes. Eles são os principais responsáveis pela viralização de conteúdo de baixa credibilidade no *X*. Além disso, esses usuários tendem a usar linguagem mais tóxica do que os demais, o que indica uma relação entre discurso agressivo e alto engajamento, sugerindo que a retórica inflamatória pode ser usada de forma estratégica para ampliar o alcance de desinformações.

Apesar das contribuições significativas do estudo para a compreensão do papel dos *superspreaders* na disseminação de conteúdos de baixa credibilidade, algumas limitações importantes podem ser destacadas. O trabalho não utiliza métodos de inteligência artificial para a classificação dos *superspreaders*, optando por métricas simples como o índice *h* e a influência, que embora eficazes, tornam o processo mais manual e dependente de análises quantitativas diretas. Essa abordagem reduz a escalabilidade e pode não capturar nuances mais complexas da rede de desinformação, como padrões dinâmicos de comportamento ou interações contextuais.

3.2 Behavioral Forensics in Social Networks: Identifying Misinformation, Disinformation and Refutation Spreaders Using Machine Learning

O estudo de Khan *et al.* (2023) apresenta uma abordagem focada na identificação de tipologias distintas de disseminadores: aqueles que propagam desinformação de forma deliberada, os que o fazem inadvertidamente, e os agentes que promovem refutações das informações falsas. A investigação sublinha a necessidade de compreender os padrões comportamentais dos usuários quando expostos a informações espúrias e suas respectivas contestações, reconhecendo que as redes sociais constituem um ambiente privilegiado para a circulação de conteúdos desinformativos.

Os autores criaram um modelo baseado em forense comportamental para classificar usuários em cinco categorias, conforme suas reações à desinformação e à sua refutação. Utilizando técnicas de aprendizado de máquina (como regressão logística e *Naive Bayes*) e *embeddings* de grafos (como Large-scale Information Network Embedding (LINE) e *PyTorch-BigGraph*), o modelo analisou a estrutura das redes sociais, padrões temporais e interações entre usuários. Isso permitiu diferenciar disseminadores intencionais e não-intencionais e proporcionou uma análise mais profunda das relações entre usuários e o conteúdo compartilhado.

Os resultados experimentais demonstraram que o modelo proposto foi capaz de alcançar uma precisão de 77,45% e um *recall* de 75,80% na detecção de atores maliciosos em um conjunto de dados extraído do *X*. A análise revelou que a maioria dos disseminadores identificados eram usuários ingênuos que corrigiam seus erros após serem expostos à refutação. Esse achado sugere que grande parte da disseminação de desinformação ocorre sem intenção maliciosa, mas sim devido à falta de verificação prévia das informações compartilhadas.

Apesar dos resultados positivos, o estudo apresenta limitações importantes. Uma delas é a dificuldade de lidar com usuários pouco ativos, o que reduz a eficácia do modelo preditivo

para esse grupo. Além disso, como o modelo foi desenvolvido com base nas características do *X*, sua aplicação em outras plataformas como o WhatsApp, que possui dinâmica comunicacional privada e estrutura de rede diferente, pode ser limitada.

3.3 Fake news spreader detection using trust-based strategies in social networks with bot filtration

O estudo de Rath *et al.* (2022) propõe uma abordagem voltada à detecção antecipada de usuários propensos a disseminar desinformação, combinando análises de confiança interpessoal e estrutura das redes sociais. Os autores destacam que, além da verificação de conteúdo, é fundamental identificar os disseminadores para conter a propagação de informações falsas, especialmente em contextos como a pandemia de COVID-19. A principal contribuição está na capacidade preditiva do modelo, permitindo agir antes que a desinformação alcance comunidades altamente conectadas, ampliando as possibilidades de mitigação.

O *framework* proposto emprega GNN em um modelo de aprendizagem indutiva, caracterizado pela adaptabilidade à redes em evolução dinâmica. A confiança interpessoal é quantificada mediante métricas específicas: *trustingness* (propensão individual a confiar em terceiros) e *trustworthiness* (probabilidade de ser percebido como confiável), derivadas da topologia da rede e do histórico de interações entre usuários. Complementarmente, o algoritmo *Trust in Social Media* computa escores de credibilidade para as arestas da rede, enquanto o *Community Health Assessment Model* identifica nós vulneráveis em comunidades através da categorização em vizinhos, fronteira e núcleo estrutural.

O modelo demonstrou acurácia de 93,3% na identificação de *spreaders* de *fake news*. A filtragem de *bots* aumentou o desempenho em até 4,6%, evidenciando que contas automatizadas distorcem dinâmicas de confiança. Estratégias baseadas em topologia foram mais eficazes que as baseadas em atividade, indicando que dados temporais enriquecidos melhoram a captura de padrões.

Apesar da robustez metodológica dos resultados, o estudo apresenta limitações importantes. A mais significativa diz respeito à forte dependência de dados históricos de atividade dos usuários. Essa característica pode comprometer seriamente a eficácia do modelo, sobretudo em perfis com baixa interação ou em contextos submetidos à políticas de privacidade mais restritivas, nos quais a disponibilidade de informações é reduzida. Esse ponto representa um desafio central para a generalização dos resultados, especialmente em plataformas como o

WhatsApp, onde as interações são privadas e protegidas por criptografia, tornando a aplicabilidade da abordagem ainda mais incerta.

3.4 FakeWhatsApp.BR: Detecção de Desinformação e Identificação de Desinformadores em Grupos Públicos do WhatsApp no Brasil

O estudo de Cabral *et al.* (2023) concentra-se em uma problemática de crescente relevância no cenário comunicacional brasileiro: o papel preponderante do WhatsApp enquanto vetor de propagação de conteúdo desinformativo, fenômeno que se intensifica significativamente durante conjunturas de elevada tensão social, como processos eleitorais presidenciais e emergências sanitárias, exemplificadas pela crise pandêmica da COVID-19. Considerando a posição central ocupada por esta plataforma no ecossistema midiático nacional, os autores propõem a construção de um corpus classificado que estabelece distinções categoriais entre mensagens caracterizadas como desinformação e aquelas de natureza verídica, fornecendo assim subsídios empíricos para o aprimoramento de algoritmos de aprendizado de máquina destinados tanto à detecção automatizada de conteúdo falso quanto à identificação de perfis responsáveis pela disseminação sistemática de desinformação.

O dataset *FakeWhatsApp.Br* foi concebido com o propósito de suprir uma lacuna identificada na literatura acerca da desinformação disseminada via *WhatsApp*, com ênfase especial no contexto das eleições presidenciais brasileiras de 2018. Por reunir dados coletados em um contexto de crise e seguir diretrizes éticas de coleta, o repositório revela-se particularmente adequado para uma análise voltada à identificação de disseminadores de desinformação no ambiente do *WhatsApp*.

A coleta dos dados foi realizada por meio de uma conta de *WhatsApp* que ingressou em 59 grupos públicos relacionados à campanha política. Esses grupos foram encontrados por meio de buscas na web e no *Facebook*, utilizando links públicos. A conta permaneceu inativa nesses grupos entre julho e novembro de 2018, capturando mensagens por meio de exportação em formato de texto plano. Ao final, os dados brutos consistiram em 59 arquivos de texto correspondentes aos grupos monitorados.

A etapa subsequente consistiu na normalização dos dados, a fim de estruturá-los em um formato tabular. Ao término desse procedimento, o conjunto de dados final contemplava 282.601 mensagens, provenientes de 5.364 usuários distribuídos por diferentes estados brasileiros.

A metodologia do estudo combina técnicas avançadas de Processamento de Linguagem Natural (PLN) com a análise de atributos comportamentais e sociais para identificar disseminadores de desinformação no WhatsApp. Utiliza o modelo Term Frequency-Inverse Document Frequency (TF-IDF) para extrair e quantificar características textuais, considerando tanto o conteúdo semântico quanto os padrões temporais das mensagens. A abordagem também leva em conta as particularidades do português brasileiro, o que aprimora a precisão dos algoritmos. Além disso, incorpora variáveis como frequência de postagens, estrutura das redes sociais e potencial de viralização do conteúdo.

A investigação identificou padrões específicos ligados à disseminação de desinformação, especialmente nas dimensões temporal e linguística, que se mostraram fortemente relacionados ao perfil dos disseminadores. Os algoritmos que consideram atributos comportamentais e sociais apresentaram bom desempenho na identificação desses agentes, com destaque para o modelo de regressão logística, que obteve acurácia de 0,997. Esse resultado reforça a eficácia da metodologia na detecção de usuários envolvidos na propagação sistemática de desinformação.

Apesar dos avanços na detecção automática de desinformação, o estudo apresenta limitações relevantes. A principal delas é a ausência de integração entre a análise textual e os metadados sociais, o que pode comprometer a eficácia na identificação dos disseminadores em cenários de maior complexidade. Diante disso, futuras pesquisas podem explorar abordagens híbridas que combinem aprendizado profundo com análise de redes sociais, ampliando a compreensão sobre a dinâmica de propagação da desinformação e o comportamento dos agentes envolvidos.

3.5 Análise Comparativa

A Tabela 3 apresenta uma análise sistemática dos quatro estudos sobre detecção de desinformação e identificação de disseminadores em plataformas digitais, evidenciando a evolução metodológica e a diversificação de abordagens na área.

Conforme demonstrado na análise individual dos artigos examinados, os estudos apresentam limitações metodológicas substanciais, particularmente no que se refere à concentração exclusiva na plataforma *X*. É possível identificar, de forma particular, os desafios metodológicos inerentes à análise comportamental de usuários no contexto do *WhatsApp*, decorrentes fundamentalmente da natureza privada e criptografada desta plataforma de comunicação.

Tabela 3 – Análise Comparativa dos Trabalhos Relacionados

Artigo	**DeVerna *et al.* (2024)**	**Khan *et al.* (2023)**	**Rath *et al.* (2022)**	**Cabral *et al.* (2023)**
Plataforma	*X*	*X*	*X*	*WhatsApp*
Foco Principal	Identificação de *superspreaders*	Detecção de tipologias distintas de disseminadores	Detecção proativa de *spreaders*	Detecção de desinformação e desinformadores em grupos públicos brasileiros
Metodologia	Métricas quantitativas (índice h, influência) + análise qualitativa	*Machine Learning* com *embeddings* de grafos	*Graph Neural Networks* + métricas de confiança	Processamento de Linguagem Natural (TF-IDF) e modelos lineares de redes neurais
Classificação de Usuários	*superspreaders* vs. usuários comuns	5 categorias comportamentais	Spreaders vs. não-spreaders	Desinformadores vs. usuários não desinformadores
Idioma	Inglês	Inglês	Inglês	Português brasileiro

Fonte: Próprio autor

Nosso estudo distingue-se por focar na identificação de disseminadores de desinformação no *WhatsApp*, empregando redes neurais para grafos. Propomos um modelo de classificação de usuários, em português brasileiro, capaz de distinguir entre disseminadores e não disseminadores com base em padrões estruturais e de interação presentes na rede.

4 METODOLOGIA

Neste capítulo, é apresentada a proposta deste trabalho para a identificação de desinformadores em grupos públicos do *WhatsApp* usando técnicas de aprendizado profundo e aprendizado de máquina. As etapas serão descritas com o objetivo de proporcionar o entendimento das técnicas aplicadas para a construção da hipótese investigativa.

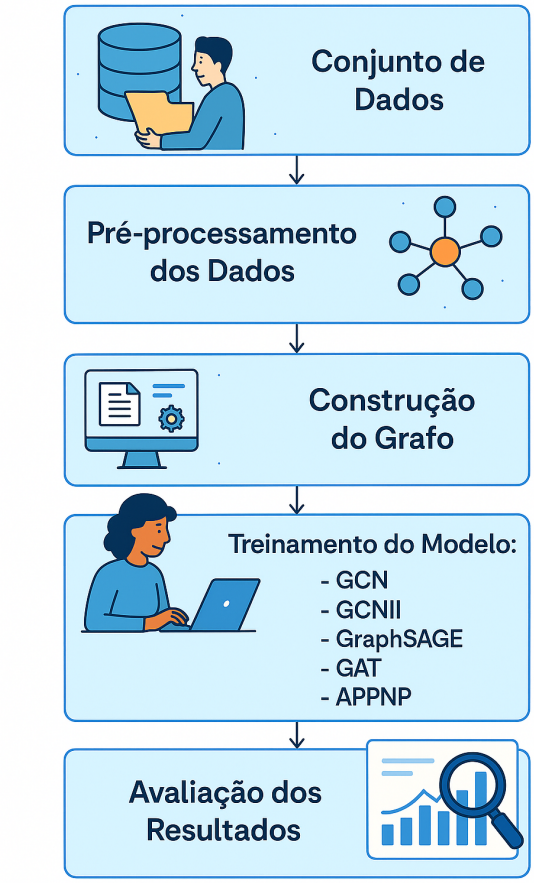
4.1 Problema e proposta

Conforme abordado ao longo do presente trabalho, o *WhatsApp* configura-se como a rede social de comunicação com maior penetração no Brasil, representando um veículo amplamente utilizado para o consumo de notícias e de informações diversas (MORENO *et al.*, 2021). Ademais, devido à natureza privada das comunicações e ao uso de criptografia, a plataforma potencializa os riscos associados à circulação de notícias falsas sobre temas sensíveis, como saúde, processos eleitorais e segurança pública (MACHADO *et al.*, 2019). Nesse contexto, a mitigação da disseminação de desinformação no *WhatsApp* revela-se uma necessidade premente, tendo em vista o papel central que a plataforma desempenha na comunicação social brasileira.

Assim, o presente trabalho tem como objetivo identificar e classificar agentes disseminadores de desinformação na plataforma *WhatsApp*, por meio da aplicação de técnicas baseadas em redes neurais para grafos. A proposta fundamenta-se na hipótese de que indivíduos responsáveis pela disseminação de informações falsas apresentam padrões de interação e difusão distintos em relação aos demais usuários. Ao explorar as conexões e interações em um grafo, a metodologia busca identificar características comportamentais indicativas de *outliers*, aptas a revelar agentes ativos na propagação de desinformação.

O modelo proposto segue as seguintes etapas: inicialmente, obtém-se o conjunto de dados brutos; em seguida, realiza-se o pré-processamento (incluindo limpeza, normalização e seleção de atributos); posteriormente, a partir desses dados preparados, constrói-se a estrutura de grafo, definindo nós e arestas. A etapa seguinte consiste no treinamento de diferentes arquiteturas de redes neurais para grafos sobre o grafo resultante. Por fim, os resultados dos modelos são avaliados por meio de métricas, como acurácia e F1-score, para determinar qual abordagem apresenta o melhor desempenho. A Figura 14 ilustra, de cima para baixo, o fluxograma do modelo proposto.

Figura 14 – Fluxograma do modelo



Fonte: Próprio autor

4.2 Conjunto de dados

O conjunto de dados empregado nesta pesquisa tem origem na dissertação de mestrado de Cabral *et al.* (2023). O arquivo de maior relevância coletado é o ***users***, disponível no repositório do conjunto de dados *FakeWhatsApp.Br*[1], o qual contém informações detalhadas acerca dos 5.364 usuários. A partir deste arquivo, elaborou-se um conjunto denominado ***users_selected_features***, que agrega as informações mais pertinentes sobre os usuários, alinhando os dados com o contexto do *WhatsApp*. Em outras palavras, foram considerados apenas os dados que podem ser extraídos diretamente da plataforma. Dessa forma, variáveis como a quantidade de mensagens contendo desinformação foram excluídas, uma vez que tais informações não são disponibilizadas de forma nativa pelo *WhatsApp*. Adicionalmente, foram incorporadas informações referentes à lista de grupos aos quais cada usuário pertence, bem como um rótulo que identifica se o usuário é ou não um disseminador de desinformação.

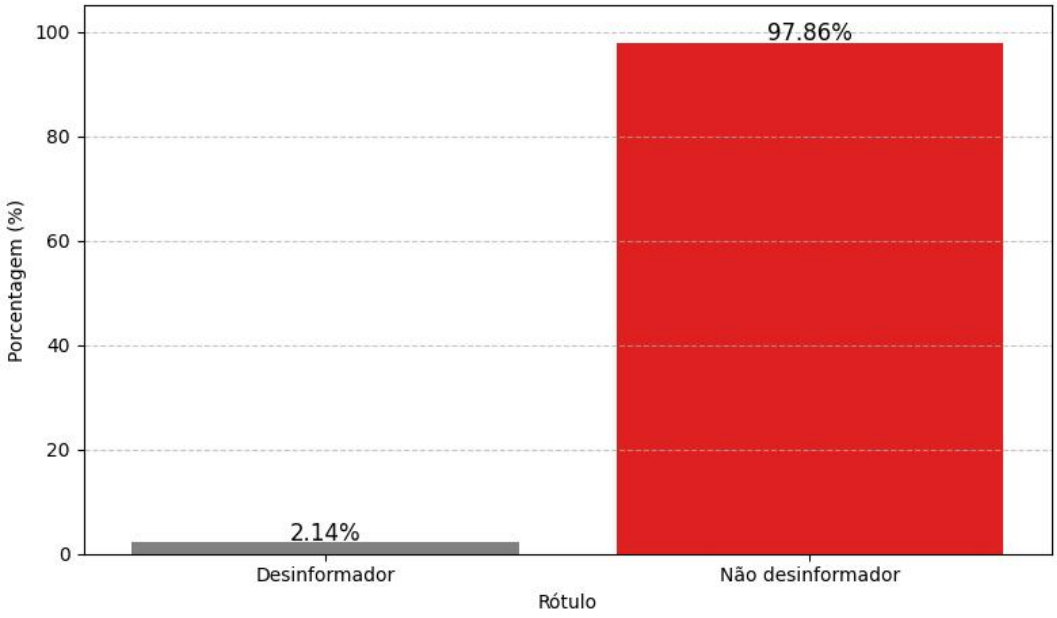
Cada usuário recebeu um rótulo conforme a classificação proposta em Cabral *et al.*

[1] https://github.com/cabrau/FakeWhatsApp.Br

(2023), a qual fundamenta-se na força viral das mensagens para identificar desinformadores. A abordagem parte da hipótese de que existe uma correlação robusta entre a disseminação de desinformação e a viralidade das mensagens. Nesse contexto, a investigação revelou que a força viral apresenta um índice de correlação de 0,87 com variáveis derivadas de dados não rotulados, sugerindo que indivíduos responsáveis pela propagação em larga escala de mensagens virais tendem, de forma acentuada, a disseminar desinformação. Em virtude disso, adota-se o critério de classificar como desinformador todo usuário cujo valor de força viral seja igual ou superior ao limiar de outlier, estabelecido empiricamente em 5.675. Dessa forma, dentre os 5.364 usuários analisados, foram classificados 132 como desinformadores.

A Figura 15 apresenta a proporção de desinformadores vs. não desinformadores. O gráfico de barras indica que 2,14% dos usuários estão classificados como desinformadores, enquanto 97,86% pertencem ao grupo não desinformador. Essa análise é representada de forma percentual, destacando o desbalanceamento considerável das classes.

Figura 15 – Proporção de Desinformadores vs. Não Desinformadores



Fonte: O autor

Essa definição de rótulos mostra-se especialmente eficaz em contextos de crise, nos quais a propagação de desinformação tende a ser mais intensa e concentrada. No entanto, sua acurácia pode não se manter tão elevada em cenários cotidianos, nos quais os padrões de compartilhamento viral assumem características menos extremas e mais difusas.

4.3 Métodos e Modelos

Nesta seção, descrevem-se os procedimentos adotados durante o desenvolvimento deste projeto. Cada fase será detalhada com o propósito de explicitar os métodos, as ferramentas e os recursos empregados.

4.3.1 Escolha dos modelos

Neste trabalho, adotam-se redes neurais em grafos (GNN) para identificar desinformadores no *WhatsApp*, devido à sua aptidão em representar dados complexos e relações estruturais das interações sociais. Ao estruturar as trocas como um grafo, capturam-se a topologia da comunicação, com nós como usuários e arestas como vínculos de mensagens. Além disso, a flexibilidade das GNN permite adaptação à variados cenários e fusão de múltiplas fontes de dados, elevando a robustez e eficácia da abordagem proposta.

Optou-se, portanto, pela aplicação de cinco modelos distintos de redes neurais para grafos: *Graph Convolutional Networks*, *Graph Convolutional Networks II*, *Graph Sample and Aggregation*, *Graph Attention Network* e *Approximate Personalized Propagation of Neural Predictions*. Cada um desses modelos foi ajustado para lidar com as especificidades dos dados e das relações entre os usuários, explorando diferentes estratégias de agregação e mecanismos de atenção.

A seleção dos cinco modelos de GNN fundamentou-se tanto na ampla adoção de cada método na literatura quanto em sua eficácia comprovada em diversas tarefas de aprendizado sobre grafos (WU *et al.*, 2020). Assim, a escolha dos modelos de GNN nesta pesquisa reflete não apenas uma estratégia metodológica robusta, mas também a adoção de técnicas que têm se mostrado promissoras em diversas áreas do conhecimento, consolidando seu papel como referência na análise de dados relacionais. Dessa forma, a adoção conjunta desses modelos garante um aporte metodológico diversificado e eficaz para a classificação de agentes disseminadores de desinformação.

A Tabela 4 apresenta uma visão resumida dos modelos selecionados para o estudo, destacando, para cada abordagem, uma breve descrição de seu funcionamento e os principais diferenciais que justificam sua adoção em tarefas de aprendizado sobre grafos.

Tabela 4 – Modelos de GNN Selecionados

Modelo	Descrição	Destaques
GCN	*Baseline* simples e eficaz para grafos médios.	Fácil implementação e bom desempenho.
GCNII	Versão aprimorada do GCN para redes profundas.	Evita *oversmoothing* e mantém acurácia.
GraphSAGE	Agrega vizinhos com amostragem eficiente.	Alta escalabilidade.
GAT	Aplica atenção às conexões relevantes.	Melhor desempenho em grafos heterogêneos.
APPNP	Propagação com PageRank personalizado.	Reduz *oversmoothing* e *overfitting*, permitindo maior profundidade da rede.

Fonte: O autor

4.3.2 Modelagem da rede

Como apresentado no Capítulo 2, as redes podem ser modeladas sob a forma de grafos de diferentes maneiras. Na abordagem adotada, um grafo que representa uma rede social consiste em uma estrutura na qual:

- **Vértices** correspondem aos indivíduos ou entidades que compõem a rede;
- **Arestas** refletem as relações ou interações estabelecidas entre esses vértices.

Como descrito por Cabral *et al.* (2023), diferentemente de redes como *X* ou *Facebook*, onde as conexões entre usuários são explícitas, no *WhatsApp* precisamos inferi-las a partir da participação em grupos. Para isso, podemos construir grafos dirigidos e ponderados em que cada nó é um usuário e cada aresta reflete o volume de mensagens enviadas (seja geral, viral ou contendo desinformação) de um usuário a outro. A Tabela 5 apresenta uma visão sintética das diferentes categorias de grafos de mensagens, detalhando para cada categoria a definição, os critérios de construção e as principais métricas estruturais.

No presente trabalho, o foco estará na construção e análise de grafos de mensagens gerais, pois o objetivo é realizar uma análise de rede que se aproxime dos contextos reais.

Com o intuito de garantir maior aderência ao ambiente operacional do *WhatsApp* e contemplar cenários realistas de uso, optou-se por manter exclusivamente as características passíveis de obtenção diretamente por meio do próprio aplicativo e da análise da estrutura da rede representada pelo grafo construído, sem a utilização de extensões ou ferramentas externas. Essa abordagem assegura que as variáveis empregadas reflitam unicamente informações nativamente acessíveis, conferindo ao estudo maior aplicabilidade prática.

Tabela 5 – Características e descrição dos grafos de mensagens

Grafo	**Descrição**	**Nós**	**Arestas**
Mensagens gerais	Cada nó é um usuário. Há aresta $i \to j$ se i enviou mensagem a um grupo de que j faz parte, com peso igual ao total de mensagens enviadas por i.	5.364	1.125.326
Mensagens virais	Aresta $i \to j$ existe se i enviou mensagem *viral* a um grupo de que j faz parte, ponderada pela quantidade de virais.	5.364	551.069
Mensagens com desinformação	Mesma lógica: aresta $i \to j$ se i enviou mensagem com desinformação a um grupo de que j faz parte, com peso pelo total dessas mensagens.	5.364	433.204

A Tabela 6 apresenta a descrição dos atributos utilizados. Cada atributo está acompanhado de uma definição concisa.

Tabela 6 – Descrição dos atributos utilizados

Feature	**Descrição**
`groups`	Grupos dos quais o usuário faz parte.
`number_of_messages`	Quantidade total de mensagens enviadas.
`texts`	Total de mensagens de texto enviadas.
`text_ratio`	Proporção de mensagens de texto em relação ao número total de mensagens.
`midia`	Objetos multimídia compartilhados (imagens, vídeos, áudios).
`midia_ratio`	Proporção de objetos multimídia em relação ao número total de mensagens.
`virals`	Interações classificadas como virais.
`repeated_messages`	Mensagens repetidas enviadas.
`strenght`	Intensidade das interações do usuário (indicador de engajamento).
`viral_strenght`	Grau de viralização das interações.

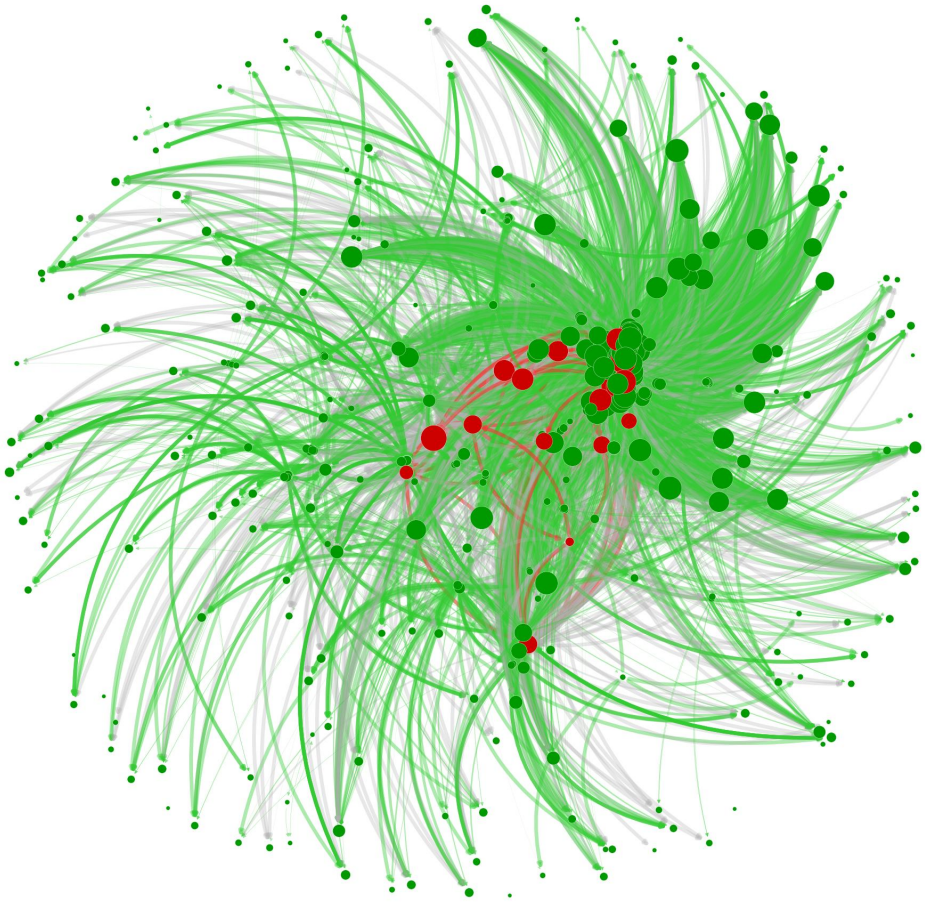
Fonte: O autor

Logo, o grafo representa usuários como nós (cada nó contendo atributos de cada usuário) e cria conexões direcionadas entre eles com base nos grupos que compartilham. A lógica de construção funciona assim: cada usuário faz parte de um ou mais grupos, e todos os usuários de um mesmo grupo são interligados por arcos (de um usuário para outro). Cada arco recebe um peso que depende da quantidade de mensagens que o usuário que origina a conexão enviou no grupo, combinado com a força viral desse usuário, refletindo o potencial de influência dele sobre os demais.

A Figura 16 apresenta uma demonstração de uma fração de 8% do grafo construído

utilizando os dados disponíveis de cada usuário. Esta rede ilustra como os usuários interagem e disseminam conteúdo em função de atributos como a `viral_strength`. Nós rotulados como desinformadores estão destacados em vermelho, enquanto os não desinformadores aparecem em verde. Usuários com maior potencial de difusão geram arestas mais espessas, representando volumes superiores de mensagens ou força de viralização elevada. Além disso, observa-se que alguns desinformadores aparecem interligados entre si por arestas vermelhas, evidenciando possíveis focos de propagação de conteúdo enganoso. As conexões entre usuários regulares surgem em outras cores, indicando um risco menor de difusão de desinformação.

Figura 16 – Demonstração de uma fração de 8% do grafo geral.



Fonte: Próprio autor

4.3.3 Implementação dos Modelos de GNN

Para a implementação dos modelos, optou-se pela linguagem `Python` devido à sua ampla gama de bibliotecas voltadas para algoritmos de aprendizado de máquina e aprendizado profundo, que oferecem suporte abrangente ao desenvolvimento e aprimoramento das técnicas

empregadas. A Tabela 7 apresenta um panorama das bibliotecas utilizadas no desenvolvimento do estudo, destacando, para cada uma, uma descrição resumida de suas funcionalidades.

Biblioteca	**Descrição Resumida**
`ast`	Análise e manipulação da árvore de sintaxe de código Python.
`numpy`	Arrays multidimensionais e operações numéricas vetorizadas.
`pandas`	Estruturas de dados tabulares (*DataFrame*) e manipulação de dados.
`networkx`	Criação, análise e visualização de grafos complexos.
`torch`	Tensores, *autograd* e construção de modelos em `PyTorch`.
`torch.nn.functional`	Funções de ativação, perdas e outras operações de camada.
`torch.optim`	Otimizadores para ajuste dos parâmetros do modelo.
`sklearn.model_selection`	Divisão de datasets em conjuntos de treino e teste.
`sklearn.metrics`	Cálculo de métricas (acurácia, precisão, *recall*, F1, matriz de confusão, *AUC-ROC*).
`sklearn.preprocessing`	Padronização de features (média 0, desvio padrão 1).
`torch_geometric.data`	Estrutura de dados para grafos em PyG (features, arestas, rótulos).
`torch_geometric.utils`	Conversão de grafos `networkx` para o formato PyG.
`torch_geometric.nn`	Camada de convolução em grafos (Graph Convolutional Network).

Tabela 7 – Resumo das bibliotecas e suas funções principais

Nossa abordagem inicia-se pela configuração do ambiente e pelo carregamento das bibliotecas necessárias. Em seguida, são realizadas as etapas já mencionadas: construção do grafo; divisão em conjuntos de treino, teste e validação; treinamento do modelo; e avaliação completa dos resultados obtidos.

4.3.3.1 Construção do grafo de mensagens

Antes de construir o grafo, define-se uma função de cálculo de peso que combina o número de mensagens trocadas por um usuário com sua força viral. Caso o valor de força viral seja zero, esse valor é substituído por um pequeno $\varepsilon = 0{,}001$ para evitar multiplicações nulas. A saída dessa função é o produto entre o número de mensagens e a força viral normalizada (dividida por 1.000):

$$\text{weight} \;=\; \text{num\_messages} \;\times\; \frac{\text{viral\_strength}}{1000} \tag{4.1}$$

O arquivo *CSV* é lido por meio de `pd.read_csv`, gerando um *DataFrame* em que cada linha representa um usuário. Cria-se então um dicionário intermediário (`groupDict`) que,

para cada identificador de grupo, armazena uma lista de tuplas (`user_id`,`message_count`), de modo que seja possível saber todos os usuários que pertencem a um mesmo grupo e quantas mensagens cada um enviou nesse contexto.

Para cada grupo em `groupDict`, acessam-se todas as tuplas $(\text{user}_i, \text{mensagens}_i)$ pertencentes aquele grupo e, em seguida, estabelecem-se arestas dirigidas entre cada par distinto de usuários ($\text{user}_i \rightarrow \text{user}_j$). Se já existir uma aresta de user_i para user_j, soma-se ao peso anterior o novo valor.

O Código-fonte 1 apresenta a construção do grafo direcionado ponderado a partir de um `CSV` de usuários e grupos. A função `CalculateWeight` normaliza a métrica `viral_strenght` (substitui zero por 0.001, senão divide por 1000) e retorna o produto com o número de mensagens, servindo como peso base de contribuição do emissor. A função `CreateMessageGraph` lê o arquivo, converte a coluna `group_list` (listas de dicionários `grupo→n_mensagens`), agrega para cada grupo a lista de (usuário, contagem), cria nós com atributos (inclusive `viral_strenght`) e adiciona arestas entre todos os pares ordenados de usuários que compartilham um mesmo grupo. O peso da aresta (`u→v`) é acumulado somando `CalculateWeight(messages_u, viral_strenght_u)` para cada grupo em comum, refletindo intensidade de participação conjunta influenciada pela força viral do emissor.

Código-fonte 1 – Construção do grafo de mensagens gerais

```
1    def CalculateWeight(num_messages, viral_strenght):
2        if viral_strenght == 0:
3            viral_strenght = 0.001
4        else:
5            viral_strenght = viral_strenght / 1000
6
7        return num_messages * (viral_strenght)
8
9    def CreateMessageGraph(file_path):
10       data = pd.read_csv(file_path)
11
12       data['group_list'] = data['group_list'].apply(ast.literal_eval)
13       viralStrengthDict = data.set_index('id')['viral_strenght'].to_dict()
14
15       groupDict = {}
16
17       for _, row in data.iterrows():
18           user_id = row['id']
19           group_list = row['group_list']
20           for group in group_list:
```

```
21                  for group_id, message_count in group.items():
22                      if group_id not in groupDict:
23                          groupDict[group_id] = []
24                      groupDict[group_id].append((user_id, message_count))
25
26          G = nx.DiGraph()
27
28          for _, row in data.iterrows():
29              node_id = row['id']
30              node_attributes = row.drop('group_list').to_dict()
31
32              G.add_node(node_id, **node_attributes)
33
34          for group_id, users in groupDict.items():
35              for i, (user_i, messages_i) in enumerate(users):
36                  for user_j, messages_j in users:
37                      if user_i != user_j:
38                          if G.has_edge(user_i, user_j):
39                              G[user_i][user_j]['weight'] += (CalculateWeight(
                                    messages_i, viralStrengthDict.get(user_i)))
40                          else:
41                              G.add_edge(user_i, user_j, weight = (
                                    CalculateWeight(messages_i, viralStrengthDict.
                                    get(user_i))))
42
43          return G
```

4.3.3.2 Particionamento do grafo em conjuntos de treino, validação e teste

São criadas partições de treino, validação e teste estratificadas por classe e, ao mesmo tempo, espalham-se nós semelhantes entre os *splits* para aumentar a diversidade topológica. Isso reduz a chance de concentrar vizinhos muito parecidos em um único conjunto, mitigando viés e redundância local. Aplica-se então o *KMeans* às características de cada nó, gerando rótulos de cluster que refletem grupos preliminares de vizinhança. Ao escolher subconjuntos para as fases de treino, validação e teste, passa-se a levar em conta não apenas a classe do rótulo binário (0 ou 1), mas também o cluster a que cada nó pertence, assegurando maior diversidade topológica e reduzindo o risco de viés.

Dentro de cada cluster, os nós são separados em subgrupos por rótulo, embaralhados aleatoriamente e divididos segundo a proporção desejada para os conjuntos de teste, validação e treino. Os índices resultantes são então convertidos em máscaras booleanas (`train_mask`,

`val_mask`, `test_mask`), marcando `True` apenas para as posições de nós selecionados. Asserções finais garantem que as máscaras sejam mutuamente exclusivas, evitando qualquer sobreposição entre os conjuntos.

4.3.3.3 Definição dos modelos GNN

Assim como explicado no Capítulo 2, GNN seguem uma estrutura semelhante à das redes neurais artificiais, com particularidades, sendo a principal delas o mecanismo de agregação de vizinhança.

O processo de aprendizado envolve múltiplas camadas em que cada nó atualiza sua representação combinando suas próprias características com as de seus vizinhos. Ao final, cada usuário no grafo possui uma representação enriquecida por informações de sua vizinhança e da estrutura global, permitindo ao modelo classificar usuários com base em seus padrões de interação e disseminação de conteúdo.

O Código-fonte 2 traz como exemplo a implementação do modelo GraphSAGE. No construtor (`init`), são instanciadas três camadas de convolução de grafos (`SAGEConv`): a primeira projeta de `input_dim` para `hidden_dim1`, a segunda de `hidden_dim1` para `hidden_dim2` e a terceira de `hidden_dim2` para `output_dim`; cada uma das duas primeiras é seguida por uma camada de normalização em lote (`BatchNorm1d`) e por um módulo de `Dropout` com taxa `dropout_rate` para mitigar overfitting. No método `forward`, os tensores de características `x` e o índice de arestas `edge_index` são processados sequencialmente: convolução, normalização, ativação (`ReLU`) e `Dropout` nas duas primeiras camadas, e somente convolução na terceira, cujo resultado final é normalizado para produzir probabilidades logarítmicas adequadas a tarefas de classificação.

Código-fonte 2 – Implementação da arquitetura GraphSAGE

```
1  class GraphSAGE(torch.nn.Module):
2      def __init__(self, input_dim, hidden_dim1, hidden_dim2, output_dim,
           dropout_rate):
3          super(GraphSAGE, self).__init__()
4
5          self.conv1 = SAGEConv(input_dim, hidden_dim1)
6          self.bn1 = torch.nn.BatchNorm1d(hidden_dim1)
7
8          self.conv2 = SAGEConv(hidden_dim1, hidden_dim2)
9          self.bn2 = torch.nn.BatchNorm1d(hidden_dim2)
```

```
10
11              self.conv3 = SAGEConv(hidden_dim2, output_dim)
12
13              self.dropout = torch.nn.Dropout(p=dropout_rate)
14
15          def forward(self, x, edge_index):
16              x = self.conv1(x, edge_index)
17              x = self.bn1(x)
18              x = F.relu(x)
19              x = self.dropout(x)
20
21              x = self.conv2(x, edge_index)
22              x = self.bn2(x)
23              x = F.relu(x)
24              x = self.dropout(x)
25
26              x = self.conv3(x, edge_index)
27
28              return F.log_softmax(x, dim=1)
```

Em síntese, ambas as variantes de arquiteturas de Redes Neurais para Grafos utilizadas neste trabalho compartilham a estrutura geral apresentada. No entanto, distinguem-se quanto à implementação interna de suas respectivas camadas de convolução, bem como na definição dos hiperparâmetros adotados.

4.4 Treinamento e Validação

O treinamento dos modelos foi conduzido utilizando-se a totalidade dos dados da base, composta por 5.364 usuários, dos quais 132 foram rotulados como desinformadores e 5.232 como não desinformadores. A fase de avaliação emprega cross-validation, um método de validação estatística empregado para avaliar o desempenho de modelos de aprendizado de máquina, especialmente em cenários com conjuntos de dados limitados. Essa técnica consiste em dividir o conjunto de dados em múltiplos subconjuntos, de modo que, em cada iteração, parte dos dados seja utilizada para o treinamento do modelo e a parte restante para a validação.

Empregou-se a técnica de validação cruzada do tipo K-Fold, pois ela proporciona uma estimativa mais confiável do desempenho do modelo, reduzindo a variância causada por uma única divisão dos dados. Ao utilizar K-Fold, o conjunto de dados é particionado em K subconjuntos mutuamente exclusivos e de dimensões equivalentes; em cada iteração, um desses

subconjuntos é reservado para teste, enquanto os K–1 restantes são usados para estimar os parâmetros do modelo.

4.4.1 Validação cruzada K-Folds

4.4.1.1 Estrutura de `StratifiedKFold`

Para garantir que cada partição reflita a mesma proporção de classes do conjunto original, utilizamos o `StratifiedKFold`. Esse procedimento divide os dados em k partições, embaralhando as amostras de forma reprodutível (`shuffle=True`, `random_state=42`). A cada iteração, uma dessas partições é reservada como teste, e as demais formam um conjunto combinado de treino e validação. Dentro desse conjunto combinado, aplicamos novamente uma separação estratificada para destinar 80% dos exemplos ao treino e 20% à validação. Assim, cada partição é usada uma única vez para teste, permitindo uma avaliação mais robusta do modelo.

4.4.1.2 Máscaras para cada partição

Em cada iteração, preparamos três vetores lógicos (booleanos) de tamanho igual ao número total de nós: `train_mask`, `val_mask` e `test_mask`. Inicialmente, todos estão com valor `False`. Depois de definir quais índices pertencem ao treino, à validação e ao teste, marcamos esses índices com `True` nas máscaras correspondentes. Essas máscaras indicam ao modelo quais nós ele deve usar em cada fase (treino, validação e teste). No final de cada partição, verificamos e imprimimos quantos nós foram alocados em cada subconjunto para assegurar o balanceamento.

4.4.1.3 Instanciação do modelo e configuração da perda

Para cada partição, criamos uma nova instância do modelo GNN atual, sem carregar nenhum peso de treinamentos anteriores. O modelo recebe como entrada o número de atributos de cada nó e retorna dois valores (já que trabalhamos com duas classes). Adotamos o otimizador `Adam`, configurado com a taxa de aprendizado e o termo de regularização `weight_decay`. Para equilibrar o impacto das classes, estimamos pesos proporcionais à frequência de cada classe em data.y e os fornecemos ao critério `NLLLoss`. Dessa forma, erros em classes menos representadas recebem uma penalidade maior, ajudando o modelo a aprender de forma balanceada.

4.4.1.4 Rotina de treinamento com early stopping

O treinamento ocorre em ciclos de épocas. A cada época, colocamos o modelo em modo de treino (`model.train()`) para ajustar os parâmetros usando apenas os nós marcados em `train_mask`. Após calcular a perda e atualizar os pesos, alternamos para modo de avaliação (`model.eval()`) sem computar gradientes para medir a perda no conjunto de validação (`val_mask`). Se essa perda for a menor observada até então, salvamos o estado atual do modelo e zeramos um contador de paciência. Caso contrário, incrementamos esse contador. Quando ele atinge o limite definido para *early stopping*, interrompemos o treinamento, pois entendemos que o modelo não está mais melhorando em dados de validação.

4.4.1.5 Avaliação final de cada partição

Depois de concluir o treinamento e recuperar o melhor estado do modelo (aquele que apresentou menor perda de validação), aplicamos o modelo aos nós de teste utilizando `test_mask`. A função `EvaluateModel(model, data, test_mask)` calcula métricas como acurácia, precisão, *recall* e F1-score para esse subconjunto. Esses resultados são armazenados em `fold_metrics` para cada partição, permitindo uma comparação individual.

4.4.1.6 Cálculo de métricas médias

Ao final de todas as k partições, dispunhamos de um conjunto de métricas para cada uma. Calculamos a média de cada métrica ao somar os valores obtidos em cada partição e dividir por k. Essa média oferece uma visão geral do desempenho do modelo, minimizando o viés que poderia surgir se avaliássemos apenas uma divisão dos dados.

5 RESULTADOS

Neste capítulo, apresentam-se e analisam-se os resultados obtidos nos experimentos de detecção de desinformação utilizando os modelos GraphSAGE, GCNII, GAT, GCN e APPNP. Os dados e os modelos utilizados estão disponíveis no repositório *Detecção de Desinformadores em Grupos do WhatsApp com GNNs*[1]

5.1 GraphSAGE: Desempenho por *Fold*

A Tabela 8 apresenta as métricas obtidas em cada um dos cinco *folds* de validação cruzada estratificada para o GraphSAGE.

Tabela 8 – Resultados do GraphSAGE por *fold* (k=5)

Fold	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*AUC-ROC*
1	0.999068	1.000000	0.961538	0.980392	0.999963
2	0.994408	0.916667	0.846154	0.880000	0.999376
3	0.998136	0.962963	0.962963	0.962963	0.999929
4	0.997204	0.961538	0.925926	0.943396	0.999752
5	0.993470	1.000000	0.730769	0.844444	0.999559
Média	**0.996457**	**0.968234**	**0.885470**	**0.922239**	**0.999716**

5.2 GCNII: Desempenho por *Fold*

A Tabela 9 apresenta as métricas obtidas em cada um dos cinco *folds* de validação cruzada estratificada para o GCNII.

Tabela 9 – Resultados do GCNII por *fold* (k=5)

Fold	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*AUC-ROC*
1	0.999068	1.000000	0.961538	0.980392	1.000000
2	0.995340	1.000000	0.807692	0.893617	0.999559
3	0.998136	0.962963	0.962963	0.962963	0.999929
4	0.997204	0.928571	0.962963	0.945455	0.999717
5	0.990672	1.000000	0.615385	0.761905	0.999301
Média	**0.996084**	**0.978307**	**0.862108**	**0.908866**	**0.999701**

[1] https://github.com/SauloCav/Deteccao-de-Desinformadores-em-Grupos-do-WhatsApp-com-GNNs

5.3 GCN: Desempenho por *Fold*

A Tabela 10 apresenta as métricas obtidas em cada um dos cinco *folds* de validação cruzada estratificada para o GCN.

Tabela 10 – Resultados do GCN por *fold* (k=5)

Fold	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*AUC-ROC*
1	0.972041	0.388889	0.269231	0.318182	0.764786
2	0.945014	0.000000	0.000000	0.000000	0.651569
3	0.974837	0.500000	0.148148	0.228571	0.669322
4	0.973905	0.000000	0.000000	0.000000	0.748778
5	0.965485	0.133333	0.076923	0.097561	0.701427
Média	**0.966256**	**0.204444**	**0.098860**	**0.128863**	**0.707176**

5.4 GAT: Desempenho por *Fold*

A Tabela 11 apresenta as métricas obtidas em cada um dos cinco *folds* de validação cruzada estratificada para o GAT.

Tabela 11 – Resultados do GAT por *fold* (k=5)

Fold	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*AUC-ROC*
1	0.946878	0.000000	0.000000	0.000000	0.716406
2	0.813607	0.046875	0.346154	0.082569	0.538829
3	0.576887	0.038877	0.666667	0.073469	0.621592
4	0.572227	0.032468	0.555556	0.061350	0.602294
5	0.911381	0.063291	0.192308	0.095238	0.719426
Média	**0.764196**	**0.036302**	**0.352137**	**0.062525**	**0.639709**

5.5 APPNP: Desempenho por *Fold*

A Tabela 12 apresenta as métricas obtidas em cada um dos cinco *folds* de validação cruzada estratificada para o APPNP.

De maneira geral, os modelos baseados em agregação de vizinhança mais robustos (GraphSAGE e GCNII) e o método de propagação personalizada (APPNP) alcançaram acurácias e *AUC-ROC* muito altas, mas exibiram discrepância entre precisão e *recall*. Esse padrão (precisão elevada combinada com *recall* mais baixo em alguns *folds*) é um forte indicativo de desequilíbrio de classes: o classificador tende a favorecer a classe majoritária, resultando em poucas falsos-

Tabela 12 – Resultados do APPNP por *fold* (k=5)

Fold	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*AUC-ROC*
1	0.986952	0.730769	0.730769	0.730769	0.991992
2	0.973905	0.472222	0.653846	0.548387	0.952942
3	0.977633	0.551724	0.592593	0.571429	0.980596
4	0.985089	0.789474	0.555556	0.652174	0.984350
5	0.984142	0.680000	0.653846	0.666667	0.979133
Média	**0.981544**	**0.644838**	**0.637322**	**0.633885**	**0.977803**

positivos (alta precisão) mas em muitos falsos-negativos (*recall* reduzido). A variabilidade entre *folds* (por exemplo, *recall* caindo muito no *fold* 5) mostra também que a generalização é sensível à divisão dos dados; embora a ordenação das previsões permaneça alta, a escolha do limiar de decisão e a distribuição de rótulos em cada *fold* impactaram fortemente métricas como *recall* e *F1*.

Os resultados muito limitados do GCN e do GAT apontam para problemas mais estruturais no ajuste desses modelos ao problema. No caso do GCN, métricas quase nulas em precisão/*recall* em alguns *folds* sugerem que o modelo colapsou para prever majoritariamente uma classe (problema comum quando a sinalização local é fraca ou quando o modelo sofre *oversmoothing* em arquiteturas profundas), ou que sua capacidade é insuficiente para separar exemplos das classes dadas as características disponíveis. Para o GAT, a instabilidade por *fold* e o baixo desempenho médio podem decorrer de atenção que não aprendeu pesos discriminativos (por exemplo por ruído nos atributos ou *hiperparâmetros* mal calibrados), tornando a aprendizagem do mecanismo de atenção ruidosa e suscetível a *overfitting*/*underfitting* dependendo da amostra de treino.

GraphSAGE e GCNII se destacaram por possírem características arquiteturais que favorecem estabilidade e preservação de informação: o GraphSAGE, por usar amostragem e agregadores locais, tende a ser mais robusto a grafos esparsos e a ruído nas vizinhanças, além de facilitar a generalização entre *folds* diferentes e reduz *overfitting* local, refletindo-se na alta *AUC* e precisão observadas; já o GCNII foi projetado para mitigar o *oversmoothing* através de conexões residuais iniciais e mapeamentos identidade, permitindo camadas mais profundas que capturam dependências de longo alcance sem colapsar as representações dos nós, o que explica sua boa capacidade discriminativa. Em conjunto, essas propriedades ajudam os modelos a aprender scores bem calibrados (*AUC* alta) e previsões positivas mais corretas.

5.6 Comparação entre os modelos

A Tabela 13 sintetiza as métricas dos modelos avaliados (GraphSAGE, GCNII, GAT, GCN e APPNP) em comparação com os resultados de limiarização e de regressão logística reportados em Cabral *et al.* (2023). Os resultados apresentados para os modelos GraphSAGE, GCNII, GAT, GCN e APPNP não correspondem às médias obtidas por *folds* na validação cruzada, mas sim à avaliação realizada a partir de uma divisão fixa dos dados em 70% para treino, 15% para validação e 15% para teste. Essa abordagem foi adotada com o objetivo de manter a consistência com os procedimentos empregados nos experimentos de limiarização e regressão logística.

Tabela 13 – Comparativo de métricas por modelo

Modelo	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*AUC-ROC*
Limiarização	0,9940	0,9160	0,8460	0,8790	—
Regressão logística	0,9970	0,9250	**0,9610**	0,9430	0,9980
GraphSAGE	**0,9972**	**1,0000**	0,9000	**0,9474**	**0,9999**
GCNII	0,9963	0,9643	0,9000	0,9310	0,9998
GAT	0,9721	0,5000	0,3000	0,3750	0,9452
APPNP	0,9764	0,6250	0,6000	0,6122	0,9777
GCN	0,8465	0,0807	0,4333	0,1361	0,7129

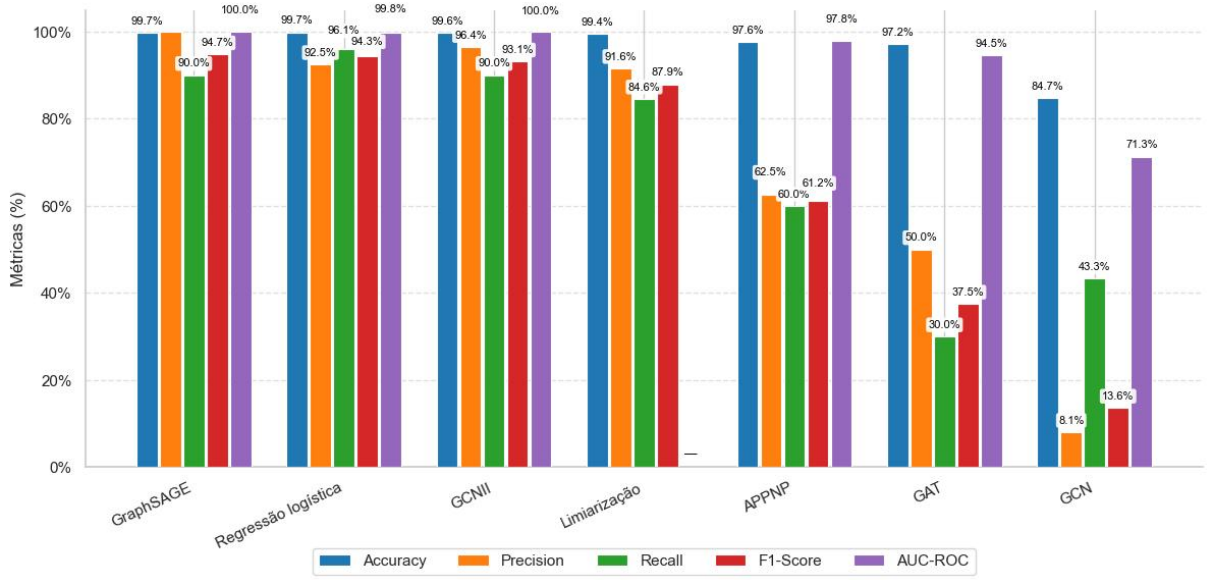
Em comparação com o trabalho de Cabral *et al.* (2023), observa-se que os modelos de GraphSAGE e GCNII, conforme mostrado na Tabela 13, apresentam desempenho igual ou superior às abordagens de limiarização e de regressão logística.

A Figura 17 apresenta um gráfico de barras comparando o desempenho dos modelos na detecção de desinformadores.

A Figura 18 apresenta um mapa de calor das métricas, comparando o desempenho dos modelos na detecção de desinformadores.

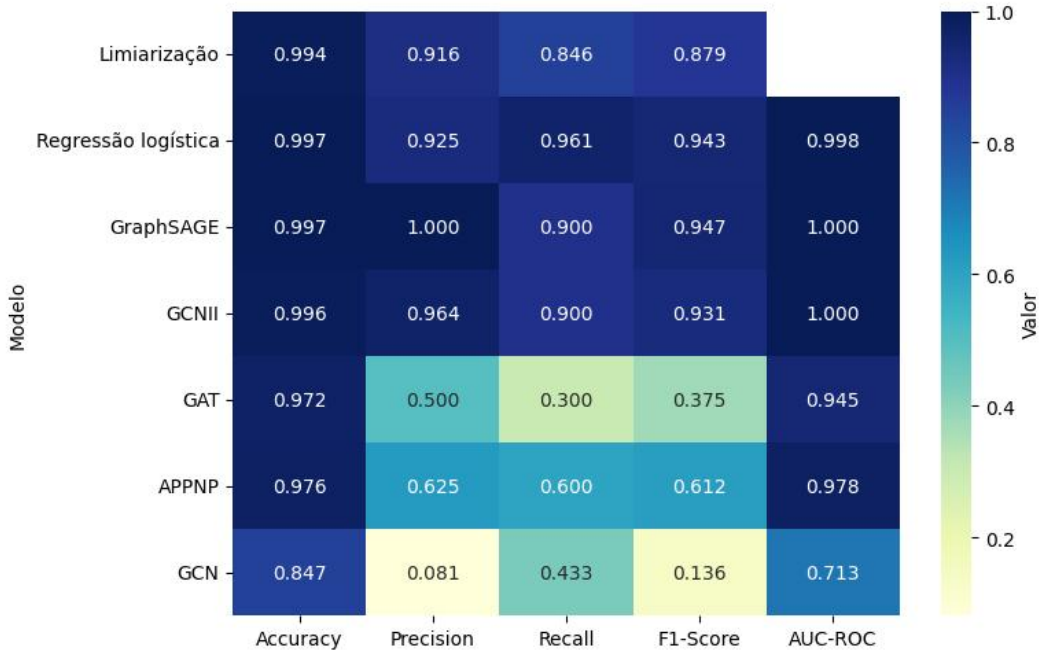
Tais resultados demonstram que as arquiteturas baseadas em grafos (em especial GraphSAGE e GCNII) não apenas mantêm elevada robustez na identificação de desinformadores, mas também apresentam melhorias consistentes em algumas das métricas-chave em relação às técnicas clássicas avaliadas por Cabral *et al.* (2023).

Figura 17 – Comparação dos modelos por meio de um gráfico de barras.



Fonte: Próprio autor

Figura 18 – Comparação dos modelos por meio de um mapa de calor das métricas.



Fonte: Próprio autor

6 CONCLUSÕES E TRABALHOS FUTUROS

Neste estudo, propôs-se e avaliou-se uma metodologia para identificar desinformadores em grupos públicos de *WhatsApp* utilizando redes neurais para grafos. Os resultados demonstraram que algumas das arquiteturas de GNN investigadas apresentam elevado potencial para capturar padrões topológicos e dinâmicas de interação viral em redes de mensagens, superando, em algumas métricas, classificadores convencionais que não exploram explicitamente a estrutura do grafo.

Em síntese, este trabalho estabeleceu uma base metodológica robusta para a aplicação de GNN na detecção de desinformadores em ambientes de mensageria, apresentando resultados promissores e indicando direções claras para aprofundamento e aplicação prática em cenários de maior complexidade.

Como trabalho futuro, sugere-se a ampliação e diversificação da base de dados utilizada, incorporando registros de diferentes períodos, idiomas e formatos de mensagem, de modo a enriquecer a representatividade dos padrões de disseminação de conteúdo. Esse acréscimo permitirá avaliar a escalabilidade do método e sua capacidade de lidar com volumes crescentes de dados.

Outro eixo relevante consiste na diferenciação entre agentes que disseminam desinformação de forma intencional (como *bots*, propagandistas e perfis coordenados) e aqueles que a veiculam inadvertidamente, ou seja, usuários comuns desinformados. A inclusão de marcadores comportamentais e semânticos específicos para cada perfil facilitará a construção de modelos mais sensíveis às motivações e estratégias de disseminação.

Por fim, é essencial analisar os dados de usuários também em períodos de baixa turbulência social. Avaliar a contribuição da força viral nesses momentos ajuda a diferenciar disseminações legítimas (mensagens amplamente compartilhadas por interesse genuíno) de estratégias deliberadas de propagação. Com isso, torna-se possível adotar esquemas de rotulagem mais sofisticados, que não se apoiem exclusivamente na força viral, mas integrem outras dimensões comportamentais e temporais para caracterizar melhor os agentes envolvidos.

REFERÊNCIAS

BAKIR, V.; MCSTAY, A. Fake news and the economy of emotions: Problems, causes, solutions. **Digital journalism**, Taylor & Francis, v. 6, n. 2, p. 154–175, 2018.

BISHOP, C. M.; NASRABADI, N. M. **Pattern recognition and machine learning**. [*S. l.*]: Springer, 2006. v. 4.

BONDY, J. A.; MURTY, U. S. R. **Graph theory**. [*S. l.*]: Springer Publishing Company, Incorporated, 2008.

BONDY, J. A.; MURTY, U. S. R. *et al.* **Graph theory with applications**. [*S. l.*]: Macmillan London, 1976. v. 290.

BORDIA, P.; DIFONZO, N. Problem solving in social interactions on the internet: Rumor as social cognition. **Social Psychology Quarterly**, Sage Publications Sage CA: Los Angeles, CA, v. 67, n. 1, p. 33–49, 2004.

BP MONEY. **Com 197 milhões de usuários no Brasil, veja como criar conexões e atrair clientes via WhatsApp**. São Paulo: [*S. n.*], 2025. Acesso em: 2 jun. 2025. Disponível em: https://bpmoney.com.br/noticias/com-197-milhoes-de-usuarios-no-brasil-veja-como-criar-conexoes-e-atrair-clientes-via-whatsapp/. Acesso em: 2025-06-02.

CABRAL, L.; MARTINS, A. D. F.; MONTEIRO, J. M.; MACHADO, J. C.; FRANCO, W. Fakespreaderswhatsapp. br: Misinformation spreaders detection in brazilian portuguese whatsapp messages. In: **ICEIS (1)**. [*S. l.*: *s. n.*], 2023. p. 219–228.

CHEN, M.; WEI, Z.; HUANG, Z.; DING, B.; LI, Y. Simple and deep graph convolutional networks. In: PMLR. **International conference on machine learning**. [*S. l.*], 2020. p. 1725–1735.

CHESNEY, R.; CITRON, D. Deepfakes and the new disinformation war: The coming age of post-truth geopolitics. **Foreign Aff.**, HeinOnline, v. 98, p. 147, 2019.

CHU, Z.; GIANVECCHIO, S.; WANG, H.; JAJODIA, S. Who is tweeting on twitter: human, bot, or cyborg? In: **Proceedings of the 26th annual computer security applications conference**. [*S. l.*: *s. n.*], 2010. p. 21–30.

CORSO, G.; CAVALLERI, L.; BEAINI, D.; LIÒ, P.; VELIČKOVIĆ, P. Principal neighbourhood aggregation for graph nets. **Advances in neural information processing systems**, v. 33, p. 13260–13271, 2020.

DEVERNA, M. R.; AIYAPPA, R.; PACHECO, D.; BRYDEN, J.; MENCZER, F. Identifying and characterizing superspreaders of low-credibility content on twitter. **Plos one**, Public Library of Science San Francisco, CA USA, v. 19, n. 5, p. e0302201, 2024.

EASLEY, D.; KLEINBERG, J. *et al.* **Networks, crowds, and markets: Reasoning about a highly connected world**. [*S. l.*]: Cambridge university press Cambridge, 2010. v. 1.

FERRARA, E.; VAROL, O.; DAVIS, C.; MENCZER, F.; FLAMMINI, A. The rise of social bots. **Communications of the ACM**, ACM New York, NY, USA, v. 59, n. 7, p. 96–104, 2016.

FERREIRA, D. A.; SILVA, A. P. da; MONTENEGRO, C. de A. O impacto das fake news na vacinação e nos surtos de doenças erradicadas. **Revista Interdisciplinar em Saúde, Cajazeiras**, v. 8, n. 8, p. 2–16, 2021.

FERREIRA, J. R. S.; LIMA, P. R. S.; SOUZA, E. D. d. Desinformação, infodemia e caos social: impactos negativos das fake news no cenário da covid-19. **Em Questão**, Universidade Federal do Rio Grande do Sul, 2021.

Fiocruz. **Pesquisa revela dados sobre fake news relacionadas à Covid-19**. Rio de Janeiro: [*S. n.*], 2020. Acesso em: 20 ago. 2025. Disponível em: https://fiocruz.br/noticia/2020/04/pesquisa-revela-dados-sobre-fake-news-relacionadas-covid-19. Acesso em: 2025-08-20.

GANSSLE, G. **Node Classification by Graph Convolutional Network**. 2018. Expero Blog. Disponível em: https://www.experoinc.com/insights/blog/node-classification-by-graph-convolutional-network.

GASTEIGER, J.; BOJCHEVSKI, A.; GÜNNEMANN, S. Predict then propagate: Graph neural networks meet personalized pagerank. **arXiv preprint arXiv:1810.05997**, 2018.

GOODFELLOW, I. **Deep learning**. [*S. l.*]: MIT press, 2016.

HAMILTON, W.; YING, Z.; LESKOVEC, J. Inductive representation learning on large graphs. **Advances in neural information processing systems**, v. 30, 2017.

HAMILTON, W. L. **Graph representation learning**. [*S. l.*]: Morgan & Claypool Publishers, 2020.

HAYKIN, S. **Neural networks: a comprehensive foundation**. [*S. l.*]: Prentice Hall PTR, 1994.

JOWETT, G. S.; O'DONNELL, V. **Propaganda & persuasion**. [*S. l.*]: Sage publications, 2018.

JR, E. C. T.; LIM, Z. W.; LING, R. Defining "fake news" a typology of scholarly definitions. **Digital journalism**, Taylor & Francis, v. 6, n. 2, p. 137–153, 2018.

KHAN, E. M.; RAM, A.; RATH, B.; VRAGA, E.; SRIVASTAVA, J. Behavioral forensics in social networks: Identifying misinformation, disinformation and refutation spreaders using machine learning. **arXiv preprint arXiv:2305.00957**, 2023.

KHEMANI, B.; PATIL, S.; KOTECHA, K.; TANWAR, S. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. **Journal of Big Data**, Springer, v. 11, n. 1, p. 18, 2024.

KIPF, T. N.; WELLING, M. Semi-supervised classification with graph convolutional networks. **arXiv preprint arXiv:1609.02907**, 2016.

LANDRUM, A. R.; OLSHANSKY, A. The role of conspiracy mentality in denial of science and susceptibility to viral deception about science. **Politics and the Life Sciences**, Cambridge University Press, v. 38, n. 2, p. 193–209, 2019.

MACHADO, C.; KIRA, B.; NARAYANAN, V.; KOLLANYI, B.; HOWARD, P. A study of misinformation in whatsapp groups with a focus on the brazilian presidential elections. In: **Companion proceedings of the 2019 World Wide Web conference**. [*S. l.*: *s. n.*], 2019. p. 1013–1019.

METZLER, H.; GARCIA, D. Social drivers and algorithmic mechanisms on digital media. **Perspectives on Psychological Science**, Sage Publications Sage CA: Los Angeles, CA, v. 19, n. 5, p. 735–748, 2024.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

MORENO, J.; NARCISO, I.; SEPÚLVEDA, R. Dinâmicas de circulação de conteúdo (des) informativo sobre a covid-19 no whatsapp, nos media e nas redes sociais online. **Dinâmicas de circulação de conteúdo (des) informativo sobre a COVID-19 no WhatsApp, nos media e nas redes sociais online.**, Observatório da Comunicação, n. 1, 2021.

PENNYCOOK, G.; BEAR, A.; COLLINS, E. T.; RAND, D. G. The implied truth effect: Attaching warnings to a subset of fake news headlines increases perceived accuracy of headlines without warnings. **Management science**, INFORMS, v. 66, n. 11, p. 4944–4957, 2020.

RATH, B.; SALECHA, A.; SRIVASTAVA, J. Fake news spreader detection using trust-based strategies in social networks with bot filtration. **Social network analysis and mining**, Springer, v. 12, n. 1, p. 66, 2022.

RUSSELL, S. J.; NORVIG, P. **Artificial intelligence: a modern approach**. [*S. l.*]: Pearson, 2016.

Rádio Senado. **Pesquisa aponta que WhatsApp é a principal fonte de informação de 79% dos entrevistados**. Brasília: [*S. n.*], 2019. Acesso em: 20 ago. 2025. Disponível em: https://www12.senado.leg.br/radio/1/noticia/2019/12/12/pesquisa-aponta-que-whatsapp-e-a-principal-fonte-de-informacao-de-79-dos-entrevistados. Acesso em: 2025-08-20.

SHU, K.; SLIVA, A.; WANG, S.; TANG, J.; LIU, H. Fake news detection on social media: A data mining perspective. **ACM SIGKDD explorations newsletter**, ACM New York, NY, USA, v. 19, n. 1, p. 22–36, 2017.

SU, Q.; WAN, M.; LIU, X.; HUANG, C.-R. *et al.* Motivations, methods and metrics of misinformation detection: an nlp perspective. **Natural Language Processing Research**, Atlantis Press, v. 1, n. 1-2, p. 1–13, 2020.

VELICKOVIC, P.; CUCURULL, G.; CASANOVA, A.; ROMERO, A.; LIO, P.; BENGIO, Y. *et al.* Graph attention networks. **stat**, v. 1050, n. 20, p. 10–48550, 2017.

VOSOUGHI, S.; ROY, D.; ARAL, S. The spread of true and false news online. **science**, American Association for the Advancement of Science, v. 359, n. 6380, p. 1146–1151, 2018.

WEST, D. B. *et al.* **Introduction to graph theory**. [*S. l.*]: Prentice hall Upper Saddle River, 2001. v. 2.

WOOD, M. J.; DOUGLAS, K. M.; SUTTON, R. M. Dead and alive: Beliefs in contradictory conspiracy theories. **Social psychological and personality science**, Sage Publications Sage CA: Los Angeles, CA, v. 3, n. 6, p. 767–773, 2012.

WOOLLEY, S. C.; HOWARD, P. N. Political communication, computational propaganda, and autonomous agents: Introduction. **International journal of Communication**, v. 10, 2016.

WU, Z.; PAN, S.; CHEN, F.; LONG, G.; ZHANG, C.; PHILIP, S. Y. A comprehensive survey on graph neural networks. **IEEE transactions on neural networks and learning systems**, IEEE, v. 32, n. 1, p. 4–24, 2020.

ZHOU, J.; CUI, G.; HU, S.; ZHANG, Z.; YANG, C.; LIU, Z.; WANG, L.; LI, C.; SUN, M. Graph neural networks: A review of methods and applications. **AI open**, Elsevier, v. 1, p. 57–81, 2020.

ZHOU, J.; LIU, L.; WEI, W.; FAN, J. Network representation learning: from preprocessing, feature extraction to node embedding. **ACM Computing Surveys (CSUR)**, ACM New York, NY, v. 55, n. 2, p. 1–35, 2022.

ZHOU, L.; BURGOON, J. K.; TWITCHELL, D. P.; QIN, T.; JR, J. F. N. A comparison of classification methods for predicting deception in computer-mediated communication. **Journal of Management Information Systems**, Taylor & Francis, v. 20, n. 4, p. 139–166, 2004.