Learning and Nonlinear Models - Journal of the Brazilian Society on Computational Intelligence (SBIC), Vol. 19, Iss. 2, pp. 69-81, 2021

UMA INTRODUÇÃO AMIGÁVEL ÀS REDES NEURAIS PARA GRAFOS

Tiago da Silva[1], **Amauri H. Souza**[2], **Diego Mesquita**[1]
[1]Fundação Getúlio Vargas, [2]Instituto Federal do Ceará
{tiago.henrique, diego.mesquita}@fgv.br, amauriholanda@ifce.edu.br

Resumo – Redes neurais para grafos (*graph neural networks*, GNNs) têm propulsionado uma série de avanços notáveis, e.g., na descoberta de novos medicamentos, na melhoria de sistemas de recomendação e na análise preditiva de redes sociais. Em essência, esses modelos extraem representações numéricas para cada nó no grafo, recursivamente combinando as representações de seus vizinhos no grafo. Este artigo tutorial apresenta alguns dos modelos de GNN mais populares e influentes, bem como discute brevemente suas aplicações em diversas áreas do conhecimento. Esperamos que este trabalho ajude a popularizar GNNs na comunidade local, fomentando o desenvolvimento científico nas áreas de aprendizado de máquina e ciência de dados.

Palavras-chave – Redes neurais para grafos, aprendizado profundo geométrico, aprendizado de máquina para grafos.

Abstract – Graph neural networks have driven a series of recent developments in, e.g., drug discovery, recommender systems, and social network analysis. At their core, GNNs are designed to extract numerical representations for each node in a graph, recursively combining representations of neighboring nodes. This tutorial paper covers some popular and influential GNN models, and discusses their applications in different disciplines. We hope this work will help popularize GNNs in the local community, and foster scientific advances in machine learning and data science.

Keywords – Graph neural networks, geometric deep learning, graph machine learning.

1. INTRODUÇÃO

Grafos são ferramentas de modelagem poderosas, com aplicações em diversas áreas do conhecimento. Químicos, por exemplo, almejam estimar os efeitos metabólicos de fármacos no tratamento de uma doença a fim de elencar um pequeno subconjunto que será submetido à processos de testes financeiramente custosos, tanto laboratoriais quanto em seres humanos [1, 2]. Criminólogos, por outro lado, estão interessados na probabilidade de que um indivíduo seja um potencial criminoso ou golpista [3, 4]. Engenheiros de tráfego comumente precisam estimar o tráfego em vias entre pares de origem e destino para otimizar estruturas viárias [5, 6]. Apesar da diversidade entre os domínios de aplicação aqui citados, esses três problemas podem ser vistos como problemas de estimar uma função sobre grafos (moléculas), sobre seus nós (indivíduos), ou sobre suas arestas (vias). No entanto, métodos clássicos de *machine learning* (e.g., multi-layer perceptrons) não conseguem incorporar a natureza relacional desses dados. Nesse caso, precisamos de métodos de regressão com maquinaria específica para nos aproveitarmos da estrutura do grafo, como é o caso das redes neurais para grafos (*graph neural networks*, GNNs).

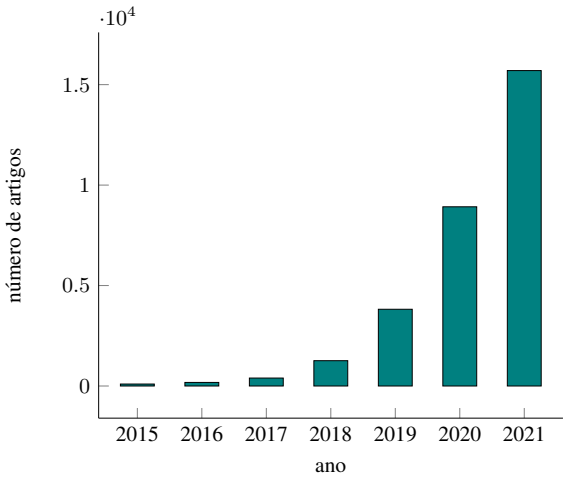


Figura 1: Popularidade de redes neurais para grafos. Números do Google Scholar com a consulta "graph convolutional networks" OR "graph neural network" OR "geometric deep learning".

Apesar da haver uma profusão de problemas preditivos em grafos, a pesquisa em GNNs ainda está na sua infância. Notavelmente, enquanto o primeiro modelo de redes neurais para grafos foi proposto por Gori et al. [7] em meados dos anos 2000, eles só se tornaram realmente populares mais que uma década depois [8–10]. Hoje, GNNs são alguns dos temas mais importantes nos veículos de publicação mais prestigiosos de aprendizado de máquina, como as conferências *advances in neural information processing systems* (NeurIPS) e a *international conference on machine learning* (ICML). A Figura 1 exibe uma estimativa do número de artigos/materiais disponíveis online sobre tópico de GNNs por ano. Podemos observar um crescimento exponencial, partindo de 98 artigos em 2015 para 15700 em 2021.

Existem dois paradigmas principais para redes neurais para grafos: GNNs baseadas em passagem de mensagem (*message passing*), e GNNs baseadas em filtros espectrais. *Message-passing* GNNs (MP-GNNs, [11]) são descritas através de um processo paralelo iterativo em que cada nó agrega informação de seus vizinhos e, subsequentemente, combina essa informação com a sua própria para atualizar seu estado (ou *embedding*). Por outro lado, GNNs espectrais (e.g., [8, 12]) surgem como um generalização

de redes neurais convolucionais para estruturas irregulares. Isso normalmente é feito redefinindo o teorema da convolução em termos da auto-decomposição do Laplaciano do grafo. Apesar da diferença conceitual entre GNNs espectrais e MP-GNNs, existem GNNs que podem ser vistas a partir de ambas as perspectivas, como *graph convolutional networks* (GCN, [9]).

Este artigo oferece uma introdução acessível às GNNs, cobrindo alguns dos desenvolvimentos mais relevantes da literatura contemporânea de GNNs. Enquanto nós focamos no paradigma de message passing (também chamado espacial), que é de longe o mais popular, também cobrimos brevemente o básico sobre GNNs espectrais. Nós discutimos também como GNNs são usadas em diferentes domínios de aplicação, bem como direções promissoras para pesquisa metodológica em GNNs. Sobretudo, nós esperamos que este trabalho fomente a pesquisa fundamental sobre GNNs em âmbito Brasileiro.

Organização do texto. Nas seções seguintes, vislumbraremos *como* GNNs efetivamente funcionam e de que maneira elas são projetadas para auxiliar o funcionamento de sistemas reais. A Seção 2 estabelece a notação que usamos no restante do artigo e caracteriza os tipos de problemas de predição sobre grafos. A Seção 3 examina como as GNNs aproveitam a topologia do grafo através da difusão de informação entre os nós em um mecanismo heuristicamente similar à passagem de mensagens em uma rede. Na Seção 4, expomos como as GNNs utilizam polinômios do Laplaciano do grafo na aprendizagem representacional dos nós e elucidamos as suas propriedades espectrais. Na Seção 5, enumeramos as aplicações reais de redes neurais para grafos em visão computacional, biomedicina, sistemas de recomendação e processamento de linguagem natural. Na Seção 6, descrevemos algumas interfaces computacionais que permitem o treinamento e a inferência de GNNs. Na Seção 7, apontamos para os principais desafios e prospectivas para o desenvolvimento de redes neurais para grafos. Na Seção 8, retrospectivamente avaliamos as considerações deste texto.

2 Preliminares

Notação. Um *grafo não direcionado* $G = (V, E)$ é caracterizado por um conjunto finito de vértices (ou nós) $V = \{1, \ldots, n\}$ e um conjunto de arestas $E \subseteq \{\{i, j\} \colon (i, j) \in V \times V\}$; escrevemos que os nós i e j estão *conectados* e que i é *vizinho* de j (e vice-versa) se $\{i, j\} \in E$. O conjunto de vizinhos de um vértice i qualquer é denotado por $\mathcal{N}(i)$.

A *matriz de adjacência* $A \in \mathbb{R}^{n \times n}$ do grafo G satisfaz $A_{ij} = \mathbb{1}[\{i, j\} \in E]$, em que simbolizamos a *função indicadora* do evento $\{i, j\} \in E$ por $\mathbb{1}[\{i, j\} \in E] = 1$ se $\{i, j\} \in E$ e $\mathbb{1}[\{i, j\} \in E] = 0$ se $\{i, j\} \notin E$. O *grau* d_i do vértice i é igual ao número de vizinhos de i, $d_i = \sum_{j \in V} A_{ji}$; nós consolidamos estes valores na *matriz diagonal de graus* D, com $D_{ii} = d_i$ e $D_{ij} = 0$ se $i \neq j$. Normalmente é utilizada a versão normalizada da matriz A com *self-loops*, definida como $\widetilde{A} = (D + I)^{-1/2}(A + I)(D + I)^{-1/2}$, em que $I \in \mathbb{R}^{n \times n}$ é a matriz identidade de tamanho n. O *Laplaciano normalizado* de G é caracterizado por $\Delta = I - D^{-1/2} A D^{-1/2}$. Em aplicações, cada nó do grafo G está tipicamente equipado com d atributos não-estruturais representados em uma matriz $X \in \mathbb{R}^{n \times d}$ (vide Figura 2). Nestes casos, nós alternativamente representamos o grafo G por $G = (A, X)$.

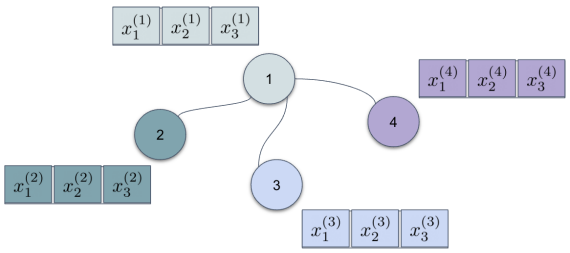
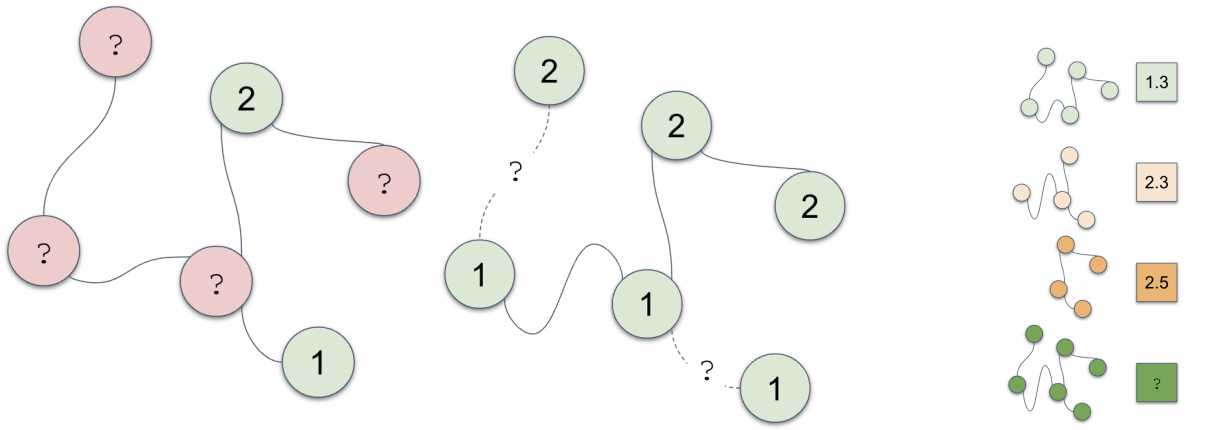


Figura 2: Exemplo de um grafo onde cada um dos $n = 4$ nós está equipado com um conjunto de $d = 3$ atributos.

Classificação de nós. Seja $\mathcal{Y} = \{1, \ldots, L\}$ um conjunto finito de classes, $G = (V, E)$ um grafo e $V_l \subseteq V$ um subconjunto de nós anotados com classes em $\mathcal{Y}$. *Classificação de nós* consiste no problema de classificar os nós em $V_l^c = V \setminus V_l$ — veja a Figura 3.a. Como exemplo, sistemas de detecção de fraude na Internet objetivam verificar a legitimidade da identidade de usuários; para isso, esses sistemas binariamente classificam como legítimo ou fraudulento os nós de uma rede de pessoas que interagem com alguma interface on-line. Existem duas diferenças essenciais entre classificação de nós em um grafo e os cenários canônicos de classificação em problemas de aprendizado supervisionado. Primeiro, supomos que o grafo, e logo seus nós/amostras, é *inteiramente observado* e que apenas não conhecemos as classes de algum subconjunto dos nós; em contraste, métodos convencionais de classificação permitem a classificação de amostras não observadas durante o treinamento do modelo — i.e., classificação de nós costuma ser uma tarefa transdutiva, enquanto métodos convencionais focam em aprendizado indutivo. Segundo, as amostras em um grafo são intrinsecamente correlacionadas e então descumprem a típica suposição de independência distribucional assumida pelos métodos historicamente relevantes de classificação – como os modelos lineares; essa inconsistência explica a efetiva inutilidade destes métodos à classificação de nós em grafos e, no passado, incentivou o desenvolvimento de procedimentos que incorporam a estrutura correlacional das amostras em seus mecanismos de inferência. Circunstancialmente, as GNNs exploram a correlação induzida nas amostras pelo seu grafo subjacente *e* as informações não estruturais para classificar os nós não anotados em um grafo.

Além disso, salientamos que o procedimento de aprendizagem de GNNs ofusca as divisas entre as categorias usuais de aprendizado supervisionado e de aprendizado não supervisionado porquanto subtemos as GNNs a uma supervisão estritamente parcial em que apenas algumas amostras estão anotadas. Tecnicamente, atestamos que as GNNs usufruem de um *aprendizado semi-supervisionado*.



(a) Classificação de nós. (b) Inferência relacional. (c) Classificação e regressão de grafos.

Figura 3: As três categorias de problemas eficazmente enfrentados por GNNs. Em classificação de nós (a), precisamos propagar as classes dos nós rotulados (em verde) aos seus vizinhos sem rótulos (em vermelho). Em inferência relacional (b), almejamos identificar a existência de uma aresta (tracejada) entre um par de nós. Em classificação e regressão de grafos (c), objetivamos atribuir classes (classificação) ou números reais (regressão) a um conjunto de grafos mutuamente independenntes em um banco de dados.

Inferência relacional (predição de aresta). Seja $G = (V, E)$ um grafo e suponha que observamos o grafo parcial $\hat{G} = (V, \hat{E})$ com $\hat{E} \subset E$; o objetivo da *inferência relacional* é identificar as arestas (*relações*) não observadas $E \setminus \hat{E}$ (Figura 3b). Por exemplo, a estimativa da probabilidade de que um par de indivíduos se conhece em uma mídia social é crucial para aumentar o engajamento dos usuários com a plataforma e corresponde a uma instanciação do problema de inferência relacional. Em outra direção, a descrição de como as diferentes proteínas interagem para permitir o desenvolvimento de um organismo é um dos problemas fundacionais de biologia molecular e é equivalente à predição de arestas no grafo de interação entre proteínas (chamado de *interatoma*). Enfaticamente, a inferência relacional, como a classificação de nós, transcende as fronteiras dos algoritmos tradicionais de aprendizagem de máquina ao exigir o tratamento de amostras correlacionadas para identificar as arestas prováveis em um espaço combinatoriamente grande de arestas possíveis. Em contraste, as GNNs eficientemente utilizam a topologia da rede e os atributos dos nós para precisamente inferir a existência de arestas de G não observadas em $\hat{G}$.

Classificação e regressão de grafos. Alguns problemas exigem o tratamento de bases de dados relacionais em que as instâncias *são* objetos representados como grafos (Figura 3c). O químico que almeja enumerar os efeitos colaterais de determinado medicamento, por exemplo, está tipicamente equipado com um conjunto de outros medicamentos com efeitos colaterais metabolicamente reconhecíveis; e cada medicamento é epistemicamente representado por uma estrutura molecular equivalente a um grafo. Esta categoria de problemas de inferência em grafos é a mais similar e receptiva à abordagem tradicional de aprendizagem de máquina; neste caso, cada grafo corresponde a uma amostra independente e presumivelmente identicamente distribuída às outras. A dificuldade incide na geração de representações vetoriais suficientemente informativas dos grafos para maximizar a eficácia de procedimentos de classificação e de regressão subsequentemente aplicados a estas representações. Notadamente, as redes neurais para grafos naturalmente aprendem representações latentes dos nós que podem ser sucessivamente agregadas e então exploradas em algoritmos de inferência canônicos de aprendizagem de máquina.

3. PASSAGEM DE MENSAGEM

Uma maneira simples de descrever o funcionamento da maioria das GNNs é através do procedimento de passagem de mensagem, na qual mensagens (vetoriais) são trocadas entre os nós que, subsequentemente, atualizam seus estados internos [11]. Em cada uma das camadas $\ell = 1 \dots L$ de uma GNN, calcula-se representações $h_v^{(\ell)} \in \mathbb{R}^{d_\ell}$ para cada nó $v \in V$ usando passagem de mensagem. A partir dos atributos de nós iniciais, $h_v^{(0)} = x_v$ $(\forall v \in V)$, as representações de nós na ℓ-ésima camada são computadas recursivamente como:

$$m_v^{(\ell)} = \textsc{Aggregate}^{(\ell)}\left(\{\!\{h_u^{(\ell-1)} : u \in \mathcal{N}(v)\}\!\}\right) \quad \forall v \in V \tag{1}$$

$$h_v^{(\ell)} = \textsc{Update}^{(\ell)}\left(h_v^{(\ell-1)}(t), m_v^{(\ell)}(t)\right) \quad \forall v \in V \tag{2}$$

Deixe que $H^{(\ell)}$ seja uma matriz contendo $h_v^{(\ell)}$ em sua v-ésima linha e que $M^{(\ell)}$ seja definida similarmente, com $m_v^{(\ell)}$ em sua

(a) Passagem de mensagem na primeira camada.

(b) Passagem de mensagem na segunda camada.

Figura 4: Passagem de mensagem em duas camadas em um único nó j. Na camada inicial (a), as representações da vizinhança deste nó são agregadas para atualizar sua representação latente. Na camada subsequente (b), incorporamos as representações dos vizinhos dos vizinhos do nó j no estágio de agregação.

v-ésima linha. Redefinindo as funções AGGREGATE e UPDATE, podemos descrever sucintamente o mesmo procedimento como

$$M^{(\ell)} = \textsc{Aggregate}^{(\ell)}\left(A, H^{(\ell-1)}\right) \tag{3}$$

$$H^{(\ell)} = \textsc{Update}^{(\ell)}\left(H^{(\ell-1)}, M^{(\ell)}\right). \tag{4}$$

Vale ressaltar que GNNs baseadas em message passing (MP-GNNs) diferem essencialmente nas funções AGGREGATE e UPDATE que elas empregam. A Figura 4 ilustra como a passagem de mensagem aplicada a um único nó funciona. A cada camada ℓ o modelo agrega mensagens dos seus vizinhos que, por sua vez, as mensagens provenientes desses vizinhos são baseadas em informações agregadas de suas respectivas vizinhanças e assim por diante. Para esse exemplo foi usado um modelo de passagem de mensagem de 2 camadas.

Utilizando $H^{(L)}$ para tarefas de predição. O procedimento acima descreve como extrair embeddings para cada nó, condensando informação sobre a estrutura do grafo ao redor do mesmo. Quando estamos interessados em classificar ou prever uma propriedade de um nó $v \in V$, usamos seu embedding final $h_v^{(L)}$ como entrada para uma rede neural feedforward (e.g., multi-layer perceptron) para obter predições da variável de saída. É importante ressaltar que as GNNs são treinadas de maneira supervisionada (*end-to-end*). Ou seja, tanto os parâmetros da GNN quanto da rede feedforward são treinados de maneira conjunta, propagando gradientes a partir de uma função de perda supervisionada (e.g., entropia cruzada ou erro quadrático médio). De forma similar, quando queremos prever uma aresta (u, v), combinamos os embeddings $h_u^{(L)}$ e $h_v^{(L)}$ através ϕ para obter um embedding de aresta $h_{uv}^{(L)}$, subsequentemente usando esse embedding como entrada para uma rede feedforward. Nesse caso, se as arestas do nosso grafo forem não-direcionadas, é importante que ϕ seja invariante a permutação, i.e., que sua saída não se altere dependendo da ordem que fornecemos os embeddings de u e v. Por exemplo, funções como média, soma e produto ponto-a-ponto são funções que obedecem a esse requisito. Por outro lado, o mesmo não ocorre para concatenação ou subtração, já que essas operações são sensíveis à ordem. Finalmente, no caso em que queremos classificar ou prever uma propriedade do grafo G, combinamos os embeddings $(h_v^{(L)})_{v\in V}$ de todos os seus nós usando uma função invariante a permutação (também chamada de *pooling* ou *readout*) para obter um embedding $h_G^{(L)}$ para o grafo. Como nos casos anteriores, usamos $h_G^{(L)}$ como entrada para uma rede feedforward que produz as predições do nosso modelo.

3.1 *Graph Convolution Network* (GCN)

O modelo *Graph Convolutional Network* (GCN, [9]) foi provavelmente o principal responsável por popularizar GNNs, possivelmente devido à sua simplicidade. A função AGGREGATE da GCN é uma soma ponderada dos vizinhos do nós $u \in \mathcal{N}(v)$, de maneira que o embedding do nó u recebe peso inversamente proporcional ao produto das raízes de $\overline{d}_u := |\mathcal{N}(u)| + 1$ e $\overline{d}_v := |\mathcal{N}(v)| + 1$. Já a função UPDATE consiste em somar o embedding de cada nó v dividido por $\overline{d}_v$ com a mensagem proveniente de seus vizinhos e aplicar uma camada linear seguida de uma função de ativação σ (tipicamente uma função *Rectified Linear Unit*, RELU). Esse processo pode ser escrito como:

$$m_v^{(\ell)} = \sum_{u\in\mathcal{N}(v)} \frac{1}{\sqrt{\overline{d}_u \overline{d}_v}} h_u^{(\ell-1)} \qquad \forall v \in V, \tag{5}$$

$$h_v^{(\ell)} = \sigma\left(\left(\frac{1}{\overline{d}_v} h_v^{(\ell-1)} + m_v^{(\ell)}\right)\Theta_\ell\right) \quad \forall v \in V, \tag{6}$$

onde $\Theta_\ell \in \mathbb{R}^{d_\ell \times d_{\ell+1}}$ é uma matriz de parâmetros/pesos. Vale ressaltar que as equações acima podem ser concisamente representadas por $H^{(\ell)} = \sigma\left(\widetilde{A} H^{(\ell-1)} \Theta_\ell\right)$. Note que GCNs utilizam uma única matriz de pesos por camada, e de forma geral, o

processamento é bem parecido com o de uma rede neural MLP, com a principal diferença sendo a etapa de difusão de embeddings de nós no grafo através da multiplicação por $\tilde{A}$.

3.2 *Graph Isomorphism Network* (GIN)

Xu et al. [13] exploram o fato de que o processo de passagem de mensagem se assemelha fortemente a uma heurística para detecção de isomorfismo em grafos chamada de teste de Weisfeiler-Lehman (WL) [14]. Nessa heurística, inicializamos os nós de ambos os grafos que queremos verificar com cores que representam seus atributos (ou uma cor padrão, caso atributos de nó não estejam disponíveis). Nos passos seguintes do teste WL, verificamos se os multisets de cores relativos a cada grafo são idênticos. Caso positivo, os nós atualizam suas cores em paralelo, aplicando uma função de HASH que toma como entrada a sua cor e o multiset com as cores de seus vizinhos. Caso contrário, o teste para indicando que os grafos não são isomórficos. Se os multisets não mudam entre duas iterações, o algoritmo encerra com resultado inconclusivo.

Note que a função HASH age de maneira análoga à composição das funções AGGREGATE e UPDATE em uma GNN, com a restrição de que HASH é uma função injetiva. Intuitivamente, então, basta que a composição AGGREGATE$\circ$UPDATE seja injetiva para que uma GNN seja tão poderosa quanto o teste WL. A fim de alcançar esse objetivo, GINs usam a seguinte implementação de *message passing*:

$$m_v^{(\ell)} = \sum_{u \in \mathcal{N}(v)} h_u^{(\ell-1)} \qquad \forall v \in V, \tag{7}$$

$$h_v^{(\ell)} = \text{MLP}^{(\ell)}\left((1+\epsilon^{(\ell)})h_v^{(\ell-1)} + m_v^{(\ell)}\right) \quad \forall v \in V, \tag{8}$$

onde $\text{MLP}^{(\ell)}(\cdot)$ representa um perceptron multicamada. Se os atributos iniciais de cada nó forem do tipo *one-hot encodings*, então podemos fazer $h_v^{(0)} = x_v$; caso contrário, utilizamos $h_v^{(0)} = \text{MLP}(x_v)$. O primeiro caso acontece pois a soma de vetores *one-hot* já é injetiva. Para mais detalhes sobre as condições que tornam GIN tão expressivo quanto o teste WL, veja a seção 4.1 do artigo original [13].

3.3 *Simple Graph Convolution* (SGC)

O modelo *Simple Graph Convolution* (SGC, [10]) é uma simplificação da rede GCN, na qual as funções de ativação das camadas internas são removidas para obter um modelo linear. Notavelmente, em diversas tarefas de classificação de nó, SGC e GCNs obtém aproximadamente a mesma performance preditiva. No entanto, por ser um modelo linear, SGC é mais interpretável e rápido. No entanto, Wu et al. [10] também apontam que SGC apresenta performance sub-ótima para tarefas à nível de grafo.

Conforme visto, as representações de nós obtidas na ℓ-ésima camada de uma GCN podem ser descritas como:

$$H^{(\ell)} = \sigma\left(\widetilde{A}H^{(\ell-1)}\Theta_\ell\right), \tag{9}$$

$$= \sigma\left(\widetilde{A}\sigma\left(\cdots\sigma\widetilde{A}\left(\widetilde{A}X\Theta_1\right)\Theta_2\ldots\right)\Theta_\ell\right), \tag{10}$$

Para reduzir o excesso de complexidade, o SGC remove as não linearidades entre as camadas da GCN, resultando em

$$H^{(\ell)} = \widetilde{A}\cdots\widetilde{A}\widetilde{A}X\Theta_1\Theta_2\cdots\Theta_\ell \tag{11}$$

$$= \widetilde{A}^\ell X\Theta_1\Theta_2\cdots\Theta_\ell \tag{12}$$

Essa simplificação torna $H^{(\ell)}$ uma transformação linear de $\widetilde{A}^\ell X$. Portanto, podemos substituir a sequência $\Theta_1, \Theta_2, \cdots, \Theta_\ell$ por uma única matriz $\Theta = \Theta_1\Theta_2\cdots\Theta_\ell$. Desta forma, a matriz de representações de nó para um SGC com ℓ camadas é dada por:

$$H = \widetilde{A}^\ell X\Theta. \tag{13}$$

3.4 *Graph Attention Network* (GAT)

A *Graph Attention Network* (GAT) [15] utiliza um mecanismo de auto-atenção (*self-attention*) para ponderar diferencialmente e adaptativamente as representações dos vizinhos de um nó na etapa de agregação da camada convolucional. Heuristicamente, esta ponderação diferencial almeja induzir a similaridade representacional entre um nó e seus vizinhos *mais relevantes*; contrastivamente, a GCN tradicional explora um protocolo de agregação agnóstico aos valores das representações/embeddings dos nós. Neste contexto, existem dois estágios computacionais em cada camada da GAT. Inicialmente, utilizamos um mecanismo de auto-atenção $a^{(\ell)}\colon \mathbb{R}^{1\times d_\ell} \times \mathbb{R}^{1\times d_\ell} \to \mathbb{R}$ compartilhado entre os nós para estimar as suas ponderações. Assim, para cada par $\{v, u\} \in E$, utilizamos

$$e_{vu}^{(\ell)} = a^{(\ell)}(h_v^{(\ell)}W_\ell, h_u^{(\ell)}W_\ell)$$

como uma quantificação da importância relacional entre as representações dos nós v e u na camada ℓ e computamos

$$\alpha_{vu}^{(\ell)} = \textsc{Softmax}(e_{v:}^{(\ell)})_u = \frac{\exp\{e_{vu}^{(\ell)}\}}{\sum_{u\in\mathcal{N}(v)}\exp\{e_{vu}^{(\ell)}\}}$$

com o objetivo de normalizar estas quantidades e garantir que elas sejam comparáveis entre diferentes vizinhanças. Em seguida, atualizamos as representações de cada nó v,

$$h_v^{(\ell+1)} = \sum_{u\in\mathcal{N}(v)} \alpha_{vu}^{(\ell)} h_u^{(\ell)},$$

em paralelo. Na prática, Veličković et al. [15] utilizaram uma composição entre uma `LeakyReLU` e uma função linear parametrizada por $\theta_\ell \in \mathbb{R}^{2d_\ell\times 1}$ como mecanismo de atenção:

$$a^{(\ell)}(x,y) = \textsc{LeakyReLU}([x\|y]\theta_\ell),$$

em que escrevemos $\|$ para o operador de concatenação. As representações da última camada são então linearmente projetadas em um espaço de dimensionalidade consistente com a tarefa supervisionada a que a rede neural está desenhada.

Notavelmente, a dependedência entre os pesos de atenção e as features iniciais dos nós tornam a GAT especialmente interessante para tarefas *indutivas*, em que a estrutura do grafo é apenas parcialmente observada durante o treinamento e a rede treinada é sucessivamente exposta à classificação ou à regressão de nós não previamente observados. Esta versatilidade permite que (1) a GAT seja aplicável em cenários em que restrições computacionais impedem o processamento de grandes grafos ou em que (2) os dados são observados sequencialmente, e o retreinamento iterativo da rede não é uma alternativa viável. Importantemente, GAT alcançou por uma margem extensa o estado da arte em classificação indutiva de nós; em problemas *transdutivos*, em que a topologia do grafo é conhecida a priori, os ganhos em acurácia desta rede são marginais e, como observado em [16], estão sensivelmente amarrados à escolha da partição da rede entre nós de treino, de validação e de teste (veja a Seção 6) – eles ocorrem ao acaso.

4. PERSPECTIVA ESPECTRAL

GNNs se popularizam com trabalhos pioneiros que buscaram generalizar operações de convolução (ou redes convolucionais) em domínios regulares (1D ou 2D) para dados em grafos, adotando uma perspectiva espectral. De forma geral, convoluções em grafos são descritas a partir dos autovalores e autovetores da Laplaciana Δ — ou de uma transformação de Δ. Como Δ é uma matriz simétrica e semidefinida positiva, sua autodecomposição é dada por $\Delta = U\Lambda U^\top$, onde $U \in \mathbb{R}^{n\times n}$ é uma matriz ortonormal com autovetores $u_1,\ldots,u_n$, e a matriz $\Lambda = \text{diag}(\lambda_1,\ldots,\lambda_n)$ contém os correspondentes autovalores (ou espectro) de Δ, com $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$.

Marjoritariamente, filtros espectrais em grafos são parametrizados por polinômios da matriz de autovalores Λ [17]. Seja $x \in \mathbb{R}^n$ um vetor coluna representando um sinal nos nós de $\mathcal{G}$. Além disso, seja $g \in \mathbb{R}^n$ um núcleo de grafo cujos coeficientes de Fourier são $\hat{g}_i = \sum_k \theta_k \lambda_i^k, i = 1,\ldots,n$. A convolução entre x e g é definida por

$$g \star x = U\text{diag}(\hat{g}_1,\ldots,\hat{g}_n)U^\top x = \sum_{k=0}^{K}\theta_k\Delta^k x, \tag{14}$$

onde $\theta_1,\ldots,\theta_K$ são parâmetros do filtro. Note que essa operação de filtragem não requer explicitamente a decomposição em autovalores da laplaciana do grafo. Por essa razão, essa abordagem também é chamada de *spectrum-free*.

De maneira similar a camadas de rede neurais convolucionais, cada camada de GNN espectral compreende: i) operações de filtragem; ii) uma soma sobre canais; e iii) uma função de ativação não-linear. Considere um sinal de grafo $X \in \mathbb{R}^{n\times d}$ com d canais e seja $H^{(0)} := X$. Então, a ℓ-ésima camada de uma GNN espectral polinomial genérica computa

$$H^{(\ell)} = \sigma\left(\sum_{k=0}^{K}\Delta^k H^{(\ell-1)}\Theta_k^{(\ell)}\right), \tag{15}$$

onde ϕ é uma não-linearidade, e $\Theta^{(\ell)} \in \mathbb{R}^{d\times d}$ são os coeficientes dos filtros espectrais.

Por exemplo, ChebNet [8] substitui os monômios sobre Δ na Equação 15 por polinômios de Chebyshev de uma laplaciana transformada $\tilde{\Delta} = (2/\lambda_n)\Delta - I$. A ideia é explorar a ortogonalidade dos polinômios de Chebyshev para obter filtros mais estáveis. Vale ressaltar que o modelo GCN foi originalmente apresentado como uma simplificação de ChebNet que utiliza filtros de primeira ordem da forma $H^{(\ell)} = \text{ReLU}\left(\tilde{A}H^{(\ell-1)}\Theta^{(\ell)}\right)$.

Trabalhos mais recentes sobre GNNs espectrais buscam melhorar ChebNets e GCNs usando filtros racionais, que podem expressar respostas de filtro mais nítidas com um número limitado de termos polinomiais. No entanto, o cálculo exato desses filtros envolve inversão de matriz, o que pode ser computacionalmente caro. CayleyNets [18] reduzem esse problema resolvendo uma sequência de sistemas lineares com o método de Jacobi. Bianchi et al. [19] propõem um design que aproxima filtros de média móvel autorregressiva com base em recursões derivadas por Isufi et al. [20].

5. APLICAÇÕES

As próximas subseções exemplificam a utilização de redes neurais para grafos em quatro diferentes cenários e advogam pela versátil inserção de GNNs em campos do conhecimento disjuntos: visão computacional, biomedicina, sistemas de recomendação e processamento de linguagem natural; em cada uma delas, os dados estão intrinsecamente dispostos em uma abstração em grafo que é adequadamente explorável pelo viés indutivo de invariância à permutação de GNNs.

5.1 Visão Computacional

Segmentação de imagens. Imagens são tecnicamente equivalentes a grafos de adjacência fixa em que identificamos os pixels com os nós e incluímos arestas entre pixels adjacentes. Circunstancialmente, as GNNs são aplicáveis em quaisquer tarefas receptivas ao tratamento espacial empreendido por redes neurais convolucionais; por outro lado, a inocuidade operativa de GNNs ao tamanho do grafo e logo à resolução da imagem também a equipam com maior flexibilidade que redes convolucionais estruturadas especificamente a imagens. Por exemplo, Liang et al. [21] instanciam um grafo baseado na distância entre pixels em uma imagem e utiliza um modelo GraphLSTM para globalmente propagar a informação dos grupos de pixels e gerar uma segmentação semanticamente consistente da imagem. Correlativamente, Landrieu and Simonovsky [22] propõem uma representação gráfica de grandes volumes de pontos em nuvem (*point clouds*) e treinam uma GNN para estimar uma segmentação semântica da cena subjacentemente representada; Qi et al. [23] executam um procedimento equivalente e treinam uma GNN desenhada para a segmentação de imagens tridimensionais em formato RGBD. Em outra direção, Kampffmeyer et al. [24] utilizam uma GNN para gerar representações vetoriais de um conjunto de classes arranjadas em um grafo de conhecimento e confrontam estas representações com as estimativas da última camada de uma ResNet [25] pré-treinada com o objetivo de atribuir imagens a classes previamente não vistas durante o treinamento[1].

Descrição cênica. A descrição de imagens em linguagem natural e – inversamente – a geração de imagens consistentes com uma descrição cênica são dois problemas enfrentados pela comunidade de visão computacional. Enfaticamente, quaisquer cenas são intrinsecamente representáveis como um grafo; os indivíduos correspondem aos nós a as suas interações correspondem às arestas. Em [26], os autores concatenam um detector de objetos e um mecanismo de passagem de mensagens para aprender a representação gráfica de um cenário exposto em uma imagem. Em contraste, Johnson et al. [27] tratam do problema inverso e utilizam uma GNN para gerar representações vetoriais da estrutura textual da descrição de uma imagem e subsequentemente exploram um protocolo adversarial para desenhar uma imagem consistente com essa descrição.

5.2 Biomedicina

Inferência de propriedades químicas. A caracterização de uma substância bioquímica como um grafo molecular em que representamos os átomos como nós e as ligações químicas como arestas enseja a utilização das versáteis redes neurais para grafos em problemas de inferência de propriedades físico-químicas, tais como os efeitos fisiológicos ou a solubilidade destas substâncias. A abordagem pioneira de [1], por exemplo, asseverou que a utilização de redes neurais convolucionais em grafos propicia o aprendizado de representações moleculares substancialmente mais informativas e estatisticamente eficientes que procedimentos anteriores amarrados a heurísticas de identificação de subestruturas moleculares específicas. Correspondentemente, Gilmer et al. [11] estimaram precisamente uma coleção de atributos quânticos de substâncias químicas utilizando uma rede neural para grafos e atestaram que apenas a estrutura topológica da molécula proporciona a inferência precisa de muitos destes atributos; isso contorna a necessidade de cômputos computacionalmente exaustivos da conformação espacial dos átomos que compõem a molécula.

Descoberta de drogas. A utilização de GNNs na exploração do espaço de medicamentos quimicamente realizáveis permite a eficiente descoberta de drogas clinicamente úteis. Em [2], os autores verificaram a eficácia de algoritmos de aprendizagem por reforço com função de recompensa parametrizada por redes neurais para grafos na descoberta de drogas com efeitos metabólicos desejáveis e com preservação de propriedades físicas cruciais (como a valência dos átomos). Mais recentemente, Bengio et al. [28] adaptaram a propriedade de conservação de massa em fluxos à parametrização por redes neurais de políticas estocásticas em espaços discretos e testemunharam a eficácia deste procedimento na descoberta de drogas com propriedades teoricamente desejáveis; em [29], os autores investigaram o uso de redes neurais para grafos na parametrização destas políticas estocásticas e sustentaram a eficácia deste procedimento em problemas de aprendizagem estrutural.

Identificação de interações entre substâncias. Proteínas são polímeros de aminoácidos com participação crucial e tipicamente catalítica nos metabolismos de organismos vivos e que executam suas funções através de uma complexa rede de interações mútuas. A identificação desta rede é essencial no tratamento de doenças que subvertam sua topologia e na compreensão das estruturas biológicas internas destes organismos. Em [30], os autores enquadram este cenário como uma instância de inferência relacional e treinam um classificador que mapeia as representações vetoriais de um par de proteínas aprendidas por uma rede neural em grafos na probabilidade de que haja interações fisiológicas entre estas proteínas. Contrastivamente, a interação entre

[1] Esta tarefa é uma instância específica do problema de *zero-shot learning*, em que exigimos inferências sobre informações não incorporadas no treinamento do algoritmo. Exemplo disso é um algoritmo treinado na classificação de árvores e que subitamente precisa distinguir palmeiras de eucaliptos.

diferentes tipos de medicamentos pode expor o organismo medicado a efeitos colaterais adversos que não seriam observados se as drogas fossem administradas individualmente; chamamos estes efeitos de *polifármacos*. A caracterização destes efeitos adversos coletivamente emergentes é moldada como um problema de inferência multirelacional, em que possibilitamos a existência de múltiplas arestas de diferentes tipos entre um par de nós. Em [31], uma rede neural para grafos é utilizada na adquirição do estado da arte em acurácia na previsão destes efeitos polifármacos.

5.3 Sistemas de recomendação

Completando matrizes. A adoção generalizada de plataformas de comércio eletrônico habilitou o registro massivo de interações entre itens de consumo e consumidores e motivou o desenvolvimento de sistemas de recomendação de mercadorias a indivíduos com o objetivo de estimular compras nestas plataformas. A representação da rede de usuários e de itens em um grafo bipartido viabiliza a utilização de redes neurais para grafos na estimativa da probabilidade de que determinado usuário aprecie determinado item. O trabalho de Monti et al. [32] recorreu a uma GNN para estimar uma representação latente da matriz de adjacência do grafo de avaliações dos usuários e iterativamente atualizou esta representação com uma rede neural recorrente; seu objetivo era aproximar o estado estacionário em que todos os itens desejáveis pelos usuários teriam sido adquiridos e utilizar esse estado na recomendação de itens ainda não comprados. No entanto, em investigação subsequente, Berg et al. [33] confirmaram empiricamente a ineficácia desta modelagem recorrente e atestaram a adequação de auto-encoders parametrizados por redes neurais para grafos na estimativa da apreciação de um item por um usuário em sistemas de recomendação. No entanto, ambos métodos exigiam disponibilidade completa do grafo em RAM e eram portanto incompatíveis com a escalabilidade exigida por sistemas de produção munidos de milhões de usuários e bilhões de itens. Nesse contexto, Ying et al. [34] propuseram um protocolo eficiente de amostragem de vizinhanças por passeio aleatório para treinamento e inferência de redes neurais convolucionais para grafos; a inclusão deste modelo como sistemas de recomendação de imagens em um site de compartilhamento de fotos com centenas de milhões de usuários atestou sua robustez.

Recomendações sociais. A interação entre usuários e a identificação de grupos sociais fortes é também um componente importante em sistemas de recomendação – pessoas socialmente próximas costumam consumir e apreciar itens similares – e a expansão das mídias sociais permitiu o acúmulo e o comércio em grande escala desse tipo de informação. Tecnicamente, a composição da rede social à rede de consumo dos usuários subverte a hipótese de bipartição e exige a exploração de métodos alternativos para explorar os atributos multirelacionais induzidos pela disponibilidade de interações sociais entre pessoas e de interações mercantis entre pessoas e objetos. Em [35, 36], os autores utilizaram a arquitetura GAT para gerar uma representação latente de itens e de pessoas e linearmente projetaram a concatenação destas representações na probabilidade de que um indivíduo aprecie um item; o objetivo da camada de atenção é heuristicamente estimar a intensidade relacional entre consumidores – que podem ser amigos próximos ou apenas colegas de trabalho com interações esporádicas.

5.4 Processamento de linguagem natural

Classificação semântica textual. A identificação da estrutura semântica subjacente de sentenças e a caracterização tópica de documentos permitem a compartimentalização informacional de corpora de textos e são assim componentes essenciais de sistemas de recuperação de informação, de mecanismos de busca e de robôs de conversação. Contextualmente, a instanciação de grafos textualmente representativos da tarefa supervisionada associada a esses corpora enseja a exploração de redes neurais para grafos em processamento de linguagem natural. Nessas circunstâncias, Marcheggiani and Titov [37] exploraram a acurácia de *parsers* sintáticos heurísticos para instanciar uma árvore sintática e sucessivamente utilizaram uma LSTM na geração de representações vetoriais dos nós desta árvore; estas representações eram então alimentadas em uma GCN que classificava a existência de alguma compatibilidade semântica entre pares de símbolos na sentença inicial. Alternativamente, os autores de [38] constroem para cada documento em um corpus um grafo textual através da inclusão de uma aresta entre palavras que ocorrem concomitantemente em uma janela de tamanho fixo e em uma mesma sentença; eles em seguida atribuem um vetor de características a cada nó deste grafo e aplicam operações convolucionais para gerar representações vetoriais utilizadas na classificação tópica destes documentos. Notavelmente, ambos os métodos redefiniram o estado da arte nas tarefas para as quais foram desenhados.

Tradução automática (*Machine Translation*). A tradução de textos entre diferentes linguagens consiste na projeção semanticamente incólume de uma sequência de palavras habitando um vocabulário a outra sequência de palavras habitando outro vocabulário. Tradução automática é um dos casos mais bem-sucedidos de aplicações de redes neurais em processamento de linguagem natural. Similarmente às instâncias de problemas de classificação textual, a representação de sentenças em árvores sintáticas e semânticas e seu subsequente tratamento com redes neurais para grafos propicia a incorporação de informações linguisticamente relevantes nas arquiteturas tipicamente recorrentes de tradução textual. Em uma abordagem inicial, Bastings et al. [39] verificaram que a construção da árvore sintática de uma sequência e sua alimentação a uma GCN também contribui substancialmente à eficiência da tradução de sentenças de inglês para alemão. Esse trabalho foi então modificado em [40] com a utilização de árvore de dependências semânticas em substituição às árvores sintáticas; mas os resultados apontaram para um incremento apenas marginal da qualidade da tradução de textos entre inglês e alemão.

Coleção	Conjunto de dados	# Grafos	Média de # nós	Média de # arestas
Benchmarking GNNs [44]	MNIST	70,000	70.57	564.53
	WikiCS	1	11,701	216,123
	PATTERN	14,000	117.47	4,749.15
	CYCLES	20,000	48.96	87.84
	ZINC	12,000	23.16	49.83
OGB [45]	proteins	1	132,534	39,561,252
	arxiv	1	169,343	1,166,243
	ddi	1	4,267	1,334,889
	wikikg2	1	2,500,604	17,137,181
	ppa	158,100	243.4	2,266.1

Tabela 1: Algumas estatísticas de conjuntos selecionados dos principais frameworks de avaliação comparativa de modelos para grafos. A biblioteca *Benchmarking GNNs* foi factualmente desenhada para avaliação da qualidade inferencial de GNNs em múltiplos grafos. Verifique a lista completa e atualizada dos conjuntos de dados disponíveis nas referências.

6. MATERIAIS E SOFTWARES

6.1 Softwares

Existem várias bibliotecas de software de código aberto disponíveis para implementar e treinar modelos de rede neural para grafos (GNN), cada uma oferecendo um conjunto único de recursos e benefícios. Atualmente, as seguintes bibliotecas são amplamente adotadas no desenvolvimento e na avaliação de redes neurais para grafos.

1. *PyTorch Geometric*[2] [41]. Uma biblioteca Python baseada no framework de aprendizado profundo PyTorch, especificamente projetada para aprendizado baseado em grafo. Essa biblioteca fornece uma implementação eficiente e flexível de modelos populares de GNN, tornando-a uma opção atraente tanto para pesquisadores quanto para profissionais.

2. *DGL*[3] [42]. A biblioteca *DGL* é outra opção popular para o desenvolvimento de GNN. Com suas capacidades escaláveis e eficientes de processamento de grafos, essa biblioteca se tornou uma solução padrão para uma série de aplicações relacionadas a grafos, desde sistemas de recomendação até a descoberta de drogas.

3. *NetworkX*[4] [43]. um pacote Python para criar, manipular e estudar redes complexas. Essa biblioteca fornece um conjunto abrangente de ferramentas para análise e visualização de grafos, tornando-a uma escolha ideal para cientistas de dados e pesquisadores de rede.

Essas bibliotecas GNN de código aberto fornecem um conjunto de ferramentas poderosas para modelagem e análise de dados de grafos complexos. Com sua arquitetura flexível e desempenho escalável, elas têm o potencial de desbloquear novos insights e descobertas em uma ampla variedade de áreas, desde biologia molecular até análise de redes sociais.

6.2 Conjuntos de dados e *benchmarking*

A popularização das redes neurais para grafos e a expansão substancial do ecossistema de arquiteturas disponíveis advogaram pelo desenvolvimento de protocolos padronizados de avaliação comparativa (*benchmarking*) em uma coleção diversa de conjuntos de dados. Enumeramos em seguida os dois principais frameworks de benchmarking de GNNs em `Python`. Ambos são compatíveis com as principais ferramentas de implementação de redes neurais para grafos – PyTorch Geometric e DGL – e concomitantemente almejam aprimorar a reprodutibilidade da pesquisa em GNNs e em propiciar uma infraestrutura consistente e adequada à experimentação de arquiteturas inéditas.

1. *Benchmarking Graph Neural Networks.*[5] A biblioteca *Benchmarking GNNs* coleciona seis conjuntos de dados reais e seis conjuntos de dados sintéticos munidos de propriedades matemáticas específicas (como grafos isomorfos ou grafos com ciclos).

2. *Open Graph Benchmarks (OGB).*[6] A biblioteca OGB contempla 14 diferentes conjuntos de dados distribuídos entre as tarefas de classificação e regressão de nós, inferência relacional e classificação e regressão de grafos.

Nós incluímos estatísticas sumárias de alguns conjuntos de dados disponíveis nestas coleções na Tabela 1.

[2]Hospedado em `https://github.com/pyg-team/pytorch_geometric`.
[3]Hospedado em `https://github.com/dmlc/dgl`.
[4]Hospedado em `https://github.com/networkx/networkx`.
[5]Hospedado em `https://github.com/graphdeeplearning/benchmarking-gnns`.
[6]Hospedado em `https://github.com/snap-stanford/ogb`.

Importantemente, estes frameworks permitem que diferentes modelos sejam justamente avaliados e que procedimentos ineditamente propostos possam ser consistentemente confrontados com as arquiteturas previamente disponíveis na literatura. Apesar disso, a pré-especificação destes procedimentos para avaliação comparativa incentivou a difusão de modelos incrementalmente mais complexos e efetivamente feitos sob medida para os conjuntos de dados contemplados nestes frameworks[7]. Shchur et al. [16] executaram experimentos de grande escala e verificaram que modificações sutis nas partições de treino, de teste e de validação destes conjuntos culminam em alterações dramáticas da acurácia relativa de diferentes modelos. Com isso, os autores atestaram a necessidade da utilização de partições aleatoriamente escolhidas.

7. DESAFIOS E NOVAS DIREÇÕES

A seguir, elencamos alguns dos principais tópicos de pesquisa e desafios em redes neurais para grafos.

Explicabilidade. Apesar do sucesso de GNNs, suas arquiteturas multi-camadas tornam suas predições difíceis de interpretar, como também ocorre em outros métodos de *deep learning*. Naturalmente, essa limitação afeta a adoção de GNNs em aplicações críticas. Mais especificamente, essa falta de interpretabilidade pode tornar mais difícil o diagnóstico de situações em que GNNs fazem predições com base em correlações espúrias ou que não estão alinhadas com o conhecimento de especialistas de domínio. Para mitigar essa limitação, uma estratégia comum é utilizar explicadores *post-hoc* — que buscam identificar elementos da entrada do modelo mais relevantes para uma dada predição. Enquanto a área de explicabilidade já é bem desenvolvida em outros domínios, há ainda um número reduzido de métodos de explicação para GNNs [46, 47]. Portanto, esperamos para os próximos anos um número significativo de novas contribuições em explicabilidade para GNNs, abrindo portas para uma adoção ainda maior de tais métodos.

Grafos temporais. Enquanto a maioria das GNNs funcionam para grafos estáticos, há um vasto número de aplicações importantes que podem ser modeladas como grafos que mudam com o tempo. Por exemplo, em uma rede social, usuários estão interagindo constantemente, gerando novas conexões e alterando seus próprios estados, bem como de outros usuários que participam de interações. Tarefas preditivas em redes sociais então correspondem a predições em grafos dinâmicos [48]. A estratégia dominante consiste em representar grafos temporais como uma sequência regular de *snapshots* [49, 50]. Mais recentemente, modelos baseados em GNNs para grafos temporais contínuos — em que interações podem acontecer a qualquer instante de tempo — foram desenvolvidos [51, 52]. Para uma revisão de modelos neurais para grafos temporais, recomendamos os trabalhos de Kazemi et al. [53] e Skarding et al. [54]. Apesar dos desenvolvimentos recentes, esperamos que novas contribuições sejam desenvolvidas nos próximos anos, especialmente para grafos temporais contínuos. Além disso, descobertas teóricos serão importantes para elucidar as limitações e o poder representacional de GNNs para grafos temporais.

Indo além do teste 1-WL. Xu et al. [13] mostraram que o poder expressivo de redes neurais baseadas em passagem de mensagem é limitado pelo teste de Weisfeiler-Lehman (WL) para isomorfismo de grafos. Mostraram ainda que a injetividade das operações de agregação e atualização é condição suficiente para obtermos máxima expressividade. Esse entendimento tem motivado o surgimento de modelos de GNN mais expressivos que 1-WL. Por exemplo, Morris et al. [55] introduziram k-GNNs que consiste GNNs de ordem k em que passagem de mensagem ocorre em um novo grafo com nós definidos como k-tuplas do conjunto de nós original. Outra forma simples de aumentar a expressividade de GNNs consiste em concatenar ruído a atributos de nós [56]. Para uma visão geral sobre expressividade de GNNs, recomendamos o trabalho de Sato [57].

Transformers para grafos. Transformer [58] tem se tornado a arquitetura dominante em várias áreas do conhecimento, principalmente em processamento de linguagem natural. Recentemente, esforços para adaptar Transformers (com *encodings* posicionais/estruturais) para tarefas de predição em grafos têm obtido resultados promissores [59]. Devido ao sucesso de Transformers, esperamos que esses recentes avanços venham incentivar o surgimento de novas arquiteturas, bem como análises teóricas sobre modelos baseados em Transformer para grafos.

8. CONSIDERAÇÕES FINAIS

O presente artigo forneceu uma visão geral sobre redes neurais para grafos. Foram introduzidas definições básicas, bem como os fundamentos dos modelos mais populares, incluindo métodos espaciais e espectrais. Além disso, foram descritas em detalhes aplicações relevantes e as ferramentas computacionais para aqueles que desejam começar a construir as primeiras redes para grafos. Por fim, as direções de pesquisa sugeridas são relativamente novas e podem ajudar a fomentar novas contribuições na área por pesquisadores da comunidade Brasileira.

É importante ressaltar que este artigo apresenta somente uma pequena fração dos avanços recentes em redes neurais para grafos. Por exemplo, em analogia com redes convolucionais para imagens, muitas GNNs empregam adicionalmente camadas de *pooling* em grafo. Visto que os benefícios dessas camadas ainda não são bem compreendidos [60], optamos por não discutir tais escolhas arquiteturais. Para uma apresentação mais detalhada sobre GNNs, recomendamos o livro proposto por Hamilton [61].

[7]Infelizmente, a regulamentação dos *benchmarks* de GNNs também predispôs uma cultura de pesquisa direcionada exclusivamente à proposta de modelos fundacionalmente frágeis e heuristicamente desestruturados com o objetivo de atingir as melhores acurácias em conjuntos de dados padronizados. A publicação de artigos que não usufruem dos melhores números tornou-se um empreendimento efetivamente inviável.

REFERÊNCIAS

[1] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," *Advances in neural information processing systems*, vol. 28, 2015.

[2] J. You, B. Liu, Z. Ying, V. Pande, and J. Leskovec, "Graph convolutional policy network for goal-directed molecular graph generation," *Advances in neural information processing systems*, vol. 31, 2018.

[3] D. Cheng, S. Xiang, C. Shang, Y. Zhang, F. Yang, and L. Zhang, "Spatio-temporal attention-based neural network for credit card fraud detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 362–369.

[4] D. Cheng, X. Wang, Y. Zhang, and L. Zhang, "Graph neural network for fraud detection via spatial-temporal attention," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[5] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2020.

[6] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

[7] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2, 2005, pp. 729–734 vol. 2.

[8] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Neural Information Processing Systems (NeurIPS)*, 2016.

[9] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[10] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International Conference on Machine Learning (ICML)*, 2019.

[11] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning (ICML)*, 2017.

[12] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," in *International Conference on Learning Representations (ICLR)*, 2014.

[13] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations (ICLR)*, 2019.

[14] B. Weisfeiler and A. Lehman, "A reduction of a graph to a canonical form and an algebra arising during this reduction," *Nauchno-Technicheskaya Informatsia*, vol. 2, no. 9, pp. 12–16, 1968.

[15] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[16] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," *arXiv preprint arXiv:1811.05868*, 2018.

[17] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond Euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[18] R. Levie, F. Monti, X. Bresson, and M. M. Bronstein, "Cayleynets: Graph convolutional neural networks with complex rational spectral filters," *IEEE Transactions on Signal Processing*, vol. 67, no. 1, pp. 97–109, 2019.

[19] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi, "Graph neural networks with convolutional ARMA filters," *arXiv preprint: 1901.01343*, 2019.

[20] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Transactions on Signal Processing*, vol. 65, no. 2, pp. 274–288, 2017.

[21] X. Liang, X. Shen, J. Feng, L. Lin, and S. Yan, "Semantic object parsing with graph LSTM," in *Computer Vision – ECCV*. Springer International Publishing, 2016, pp. 125–143.

[22] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[23] X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun, "3d graph neural networks for RGBD semantic segmentation," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[24] M. Kampffmeyer, Y. Chen, X. Liang, H. Wang, Y. Zhang, and E. P. Xing, "Rethinking knowledge graph propagation for zero-shot learning," in *IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2019, pp. 11 487–11 496.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[26] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei, "Scene graph generation by iterative message passing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[27] J. Johnson, A. Gupta, and L. Fei-Fei, "Image generation from scene graphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1219–1228.

[28] E. Bengio, M. Jain, M. Korablyov, D. Precup, and Y. Bengio, "Flow network based generative models for non-iterative diverse candidate generation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 381–27 394, 2021.

[29] T. Deleu, A. Góis, C. Emezue, M. Rankawat, S. Lacoste-Julien, S. Bauer, and Y. Bengio, "Bayesian structure learning with generative flow networks," in *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, vol. 180, 2022, pp. 518–528.

[30] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur, "Protein interface prediction using graph convolutional networks," *Advances in neural information processing systems*, vol. 30, 2017.

[31] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *Bioinformatics*, vol. 34, no. 13, pp. i457–i466, 2018.

[32] F. Monti, M. Bronstein, and X. Bresson, "Geometric matrix completion with recurrent multi-graph neural networks," *Advances in neural information processing systems*, vol. 30, 2017.

[33] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[34] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018.

[35] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The World Wide Web Conference*, 2019.

[36] Q. Wu, H. Zhang, X. Gao, P. He, P. Weng, H. Gao, and G. Chen, "Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems," in *The World Wide Web Conference*, 2019.

[37] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017.

[38] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-CNN," in *World Wide Web Conference*, 2018.

[39] J. Bastings, I. Titov, W. Aziz, D. Marcheggiani, and K. Simaan, "Graph convolutional encoders for syntax-aware neural machine translation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017.

[40] D. Marcheggiani, J. Bastings, and I. Titov, "Exploiting semantics in neural machine translation with graph convolutional networks," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, 2018.

[41] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[42] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep graph library: A graph-centric, highly-performant package for graph neural networks," *arXiv preprint arXiv:1909.01315*, 2019.

[43] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," in *Proceedings of the 7th Python in Science Conference*, G. Varoquaux, T. Vaught, and J. Millman, Eds., Pasadena, CA USA, 2008, pp. 11 – 15.

[44] V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *arXiv preprint arXiv:2003.00982*, 2020.

[45] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," *Advances in neural information processing systems*, vol. 33, pp. 22 118–22 133, 2020.

[46] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[47] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang, "Parameterized explainer for graph neural network," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[48] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[49] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *International Conference on Neural Information Processing (ICONIP)*, 2018.

[50] A. Pareja, G. Domeniconi, J. Chen, T. Ma, H. K. T. Suzumura, T. Kaler, T. B. Schardl, and C. E. Leiserson, "EvolveGCN: Evolving graph convolutional networks for dynamic graphs," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[51] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, E. Monti, and M. Bronstein, "Temporal graph networks for deep learning on dynamic graphs," in *ICML 2020 Workshop on Graph Representation Learning*, 2020.

[52] S. Kumar, X. Zhang, and J. Leskovec, "Predicting dynamic embedding trajectory in temporal interaction networks," in *International Conference on Knowledge Discovery & Data Mining (KDD)*, 2019.

[53] S. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, and P. Poupart, "Representation learning for dynamic graphs: A survey," *Journal of Machine Learning Research*, vol. 21, no. 70, pp. 1–73, 2020.

[54] J. Skarding, B. Gabrys, and K. Musial, "Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey," *IEEE Access*, vol. 9, pp. 79 143–79 168, 2021.

[55] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, "Weisfeiler and leman go neural: Higher-order graph neural networks," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[56] R. Sato, M. Yamada, and H. Kashima, "Random features strengthen graph neural networks," in *SIAM International Conference on Data Mining (SDM)*, 2021.

[57] R. Sato, "A survey on the expressive power of graph neural networks," *arXiv:2003.04078*, 2020.

[58] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[59] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu, "Do transformers really perform badly for graph representation?" in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[60] D. Mesquita, A. H. Souza, and S. Kaski, "Rethinking pooling in graph neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[61] W. L. Hamilton, "Graph representation learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 14, no. 3, pp. 1–159.