Explicabilidade de *Graph Neural Networks*: Uma Análise Comparativa com SHAP

Isabela Buzzo Galvão[1], **Anderson Bessa da Costa**[2]

[1]Faculdade de Computação – Universidade Federal de Mato Grosso do Sul (UFMS)
Caixa Postal 549 – 79.070-900 – Campo Grande – MS – Brazil

{isabela.buzzo,anderson.bessa}@ufms.br

Abstract. *Graph Neural Networks (GNNs) are effective models for learning from structured data such as graphs and are applied in diverse areas. However, their interpretability remains a challenge, especially in critical applications. Methods like SHAP (Shapley Additive Explanations) provide insights into predictions, but their application to GNNs is complex due to the relational structure of the data. This work reviews explanation approaches for GNNs, with an emphasis on those aligned with SHAP. The analysis shows that the methods, although similar in concept, differ in the choice of graph component used for the explanation among nodes, edges, or subgraphs and in the approximation techniques for calculating Shapley values. It is concluded that the choice of the appropriate method resides in the nature of the chosen task and the type of understanding sought.*

Resumo. *Graph Neural Networks (GNNs) são modelos eficazes para aprendizado em dados estruturados como grafos e são aplicadas em diversas áreas. No entanto, sua interpretabilidade ainda é um desafio, especialmente em aplicações críticas. Métodos como o SHAP (Shapley Additive Explanations) fornecem explicações sobre predições, mas sua aplicação em GNNs é complexa devido à estrutura relacional dos dados. Este trabalho revisa abordagens de explicação para GNNs, com ênfase naquelas alinhadas ao SHAP. A análise evidencia que os métodos, embora similares em conceito, diferem-se na escolha de componente do grafo utilizada na explicação dentre nós, arestas ou subgrafos e nas técnicas de aproximação para cálculo dos valores de Shapley. Conclui-se que a escolha do método apropriado reside na natureza da tarefa escolhida e no tipo de entendimento que se deseja obter.*

1. Introdução

As *Graph Neural Networks* [Scarselli et al. 2009] têm emergido como uma abordagem poderosa para o aprendizado em dados estruturados, como os grafos. Diferente dos modelos tradicionais de aprendizado profundo, que operam sobre dados tabulares ou imagens, as GNNs exploram a topologia das conexões entre as entidades de um grafo, muitas vezes complexas e dinâmicas. Esse tipo de modelo têm sido aplicado em diversas áreas, incluindo bioinformática, análise de redes sociais, extração de relações textuais e sistemas de recomendação [Réau et al. 2022, Yang et al. 2021, Wu et al. 2020, Peng et al. 2017, Fan et al. 2019] demonstrando um desempenho superior em tarefas que envolvem dados estruturados em grafos.

Entretanto, à medida que esses modelos se tornam mais complexos, a compreensão de suas predições torna-se um desafio significativo. Em muitos cenários, a explicação de um modelo é tão importante quanto o seu desempenho preditivo pois garantem transparência, confiabilidade e aceitação em aplicações críticas, como saúde e finanças. Essa importância é ainda mais crucial levando em conta o fato de que modelos de aprendizado de máquina frequentemente operam como caixas-pretas, dificultando a compreensão de suas decisões e reduzindo sua interpretabilidade. Entre as diversas abordagens de explicação, o *Shapley Additive Explanations* (SHAP) [Lundberg and Lee 2017], ainda se destaca como o estado da arte, baseando-se na teoria dos valores de Shapley e Teoria dos Jogos para quantificar a contribuição de cada componente da entrada na predição de modelos. Apesar de sua popularidade em modelos tabulares e em redes neurais convencionais, sua aplicação direta a GNNs ainda apresenta desafios, dado o caráter relacional dos dados.

A explicação de GNNs é um campo emergente de pesquisa, e diversas abordagens têm sido propostas para interpretar esses modelos. Na literatura, algumas dessas abordagens apresentam alinhamento com os princípios do SHAP, buscando fornecer explicações baseadas na contribuição de nós, arestas e subgrafos para a predição do modelo. Esse alinhamento sugere que há fundamentos compartilhados entre as explicações de modelos GNN e explicações baseadas no SHAP, o que motiva uma análise mais detalhada dessa relação. No entanto, ainda há lacunas na compreensão sobre quais aspectos dessas explicações são realmente compatíveis.

O objetivo deste trabalho é estudar, revisar e apresentar uma análise comparativa das diferentes propostas de explicação de modelos de GNN, com ênfase naquelas que se alinham ao SHAP, a fim de caracterizar o que as torna semelhantes e o que as diferencia. Investigamos quais componentes do grafo são priorizados por cada método e quais estratégias de adaptação são utilizadas para alcançar o valor de Shapley. Por fim, discutimos as limitações e o panorama de futuras pesquisas na área. O restante deste artigo está estruturado da seguinte forma: na Seção 2, apresentamos os fundamentos teóricos do modelo da GNN; na Seção 3, detalhamos o *framework* SHAP; na Seção 4, revisamos abordagens da literatura para explicações de GNNs alinhadas ao SHAP; na Seção 5, apresentamos a análise comparativa desses métodos; e na Seção 6, concluímos com as considerações finais e sugestões para pesquisas futuras.

2. O modelo da GNN

As GNNs foram introduzidas por Franco Scarselli [Scarselli et al. 2009] como uma extensão das Redes Neurais Recursivas (RNN) [Hopfield 1982] e das Redes de Markov [Markov 2006], além de introduzir a capacidade de interpretação das relações topológicas de um grafo de maneira direta e de ampliar o escopo de processamento das RNNs para uma variedade maior de grafos (tais como grafos direcionados e cíclicos).

Comumente, modelos tradicionais de aprendizado de máquina sem suporte para grafos necessitam de uma etapa de pré-processamento. Essa etapa objetiva transformar o grafo em uma estrutura simplificada onde o grafo é, primeiramente, reduzido a uma representação baseada em lista (como um vetor de reais [Belkin and Niyogi 2003], [Zhou et al. 2003], por exemplo), para, então, ser processado. No entanto, essa abordagem pode levar à perda de informações cruciais, pois não considera as dependências to-

pológicas entre nós. As GNNs, por outro lado, são capazes de manter a estrutura original do grafo, compreendendo melhor seus relacionamentos e dependências.

Formalmente, um grafo G pode ser definido como $G = (N, E)$, onde N é seu conjunto de nós (vértices), e E, seu conjunto de arestas. Nas GNNs, os nós e arestas do grafo são associados à atributos, que são representados por um vetor de reais. Esses vetores são caracterizados por $l_n \in \Re^{l_N}$ para nós e $l_{(n1,n2)} \in \Re^{l_E}$ para arestas, onde $n \in N$. Além disso, a cada nó também é atribuído um estado $x_n \in \Re^S$ onde S representa um subconjunto dos nós de G, cujo valor é baseado nas informações dos vizinhos desse nós. A partir dessas informações, o modelo é capaz de aprender representações internas que permitem classificar nós, prever a existência de novas arestas ou classificar grafos inteiros. Esse processo acontece por meio de um mecanismo iterativo de atualização de estado entre vizinhos, no qual cada nó atualiza repetidamente o próprio estado, permitindo à GNN capturar relações mais complexas e intrínsecas, propagando e trocando informações através da vizinhança inteira. Tal mecanismo assemelha-se à operação de convolução, realizada pelas *Convolutional Neural Networks* (CNNs), que utiliza uma função de janela deslizante (*kernel*) para extrair características locais a partir dos valores dos pixels vizinhos. Essencialmente, o princípio é o mesmo tratando-se da atualização de estados das GNNs: combinar informações locais para construir representações mais expressivas e informativas dos elementos da estrutura. A Figura 1 exemplifica esse mecanismo aplicado ao nó 1. A imagem detalha como o estado x_1 é calculado pela função f_w agregando os atributos l (dos nós e arestas incidentes) e os estados x da vizinhança, conforme definido anteriormente.

O mecanismo de atualização de estado da GNN pode ser matematicamente representado da seguinte forma (adotaremos neste trabalho a mesma notação do trabalho de Franco Scarselli [Scarselli et al. 2009]):

$$x_n(t+1) = f_w(l_n, l_{co[n]}, x_{ne[n]}(t), l_{ne[n]})$$

$$o_n(t) = g_w(x_n(t), l_n), n \in \mathbb{N}.$$

O estado do nó n no instante t é dado por $x_n(t)$. A notação $l_{co[n]}$ denota os atributos das arestas conectadas a n, $l_{[n]}$ denota os atributos do nó n, $x_{ne[n]}(t)$ representa o estado dos nós vizinhos de n no tempo t e $l_{ne[n]}$ refere-se aos atributos dos nós vizinhos. A função f_w é responsável pela atualização de estado local do nó, incorporando informações da vizinhança, enquanto g_w é a função de saída local, que mapeia o valor de saída do nó, representado por $o_n(t)$. O conjunto de todos os estados (x) e o conjunto de todos os valores de saída (o) devem ser únicos, sendo que o define um mapa $\varphi_w : D \rightarrow \Re^m$ que aceita um grafo como entrada e devolve uma saída o_n para cada nó. A unicidade e existência desses conjuntos é fundamentada e garantida pelo teorema do ponto fixo de Banach, desde que f_w seja um mapa de contração, algo que é garantido na implementação da GNN.

A computação do mecanismo de atualização de estados ocorre iterativamente em todos os nós do grafo e pode ser vista como uma rede formada por duas unidades, onde uma é responsável por calcular as iterações de f_w até sua convergência e a outra unidade computaria, então, a saída por meio de g_w. Quando ambas as unidades são implementadas como redes neurais *feed-forward* (FNN), a rede em questão torna-se uma rede neural

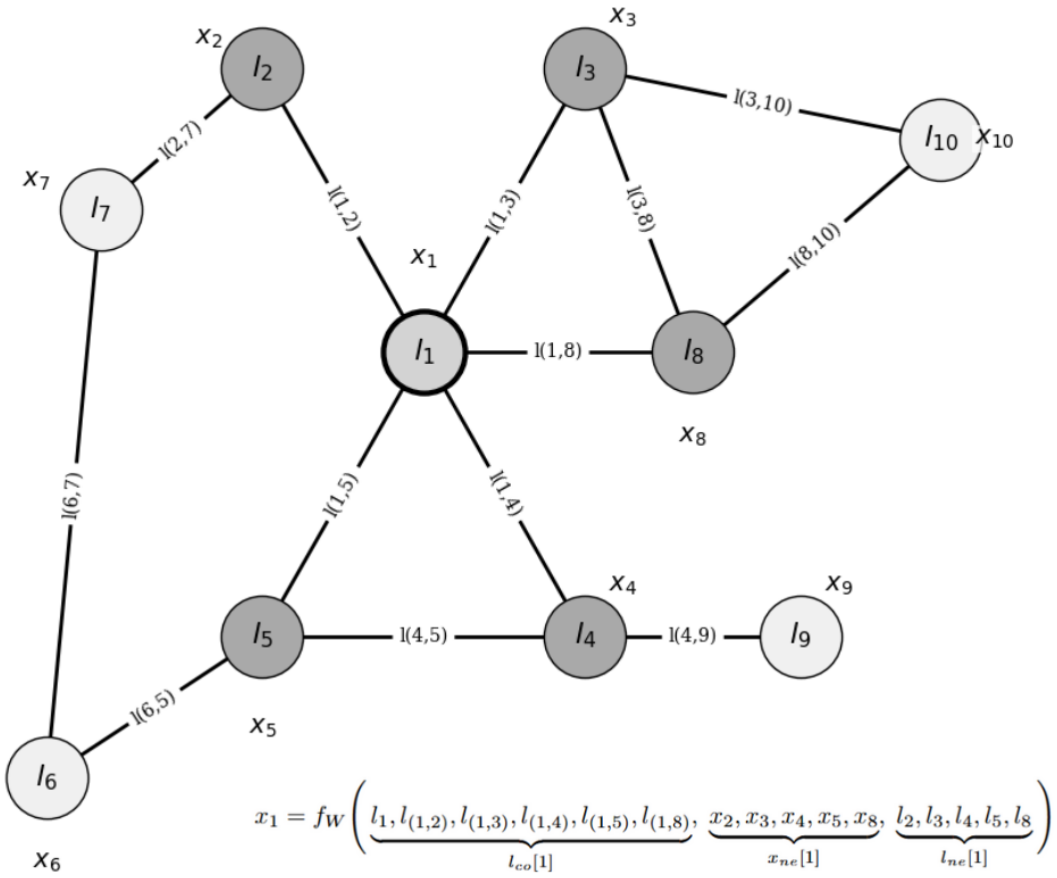

Figura 1. Representação do mecanismo de atualização de estado em uma GNN. Inspirado em Scarselli [Scarselli et al. 2009].

recursiva. Isso se dá pois as conexões entre unidades criam um fluxo de informação cíclico e dependente de iterações, algo característico das RNNs.

À medida que o estado dos nós se atualiza com base em seus vizinhos, as informações fluem através do grafo, permitindo que nós distantes compartilhem conhecimento ao longo das iterações. A convergência de f_w é de ordem exponencial, e, geralmente, 5 a 15 iterações são suficientes para alcançá-la. A fase de teste da GNN é computacionalmente eficiente: o custo de cada iteração cresce linearmente em relação ao número de arestas do grafo, à dimensão das FNNs e à dimensão do estado. Já a fase de aprendizado demanda maior tempo computacional. Embora a maioria das instruções mantenha complexidade linear em relação ao grafo e às FNNs, o cálculo do gradiente introduz exceções que dependem quadraticamente da dimensão do estado.

Assim, as GNNs introduzem um modelo eficaz para aprendizado em grafos, superando a perda de informações dos modelos tradicionais ao lidar diretamente com sua topologia de forma eficiente e contextualizada.

3. Explicações com *Shapley Additive Explanations* (SHAP)

O SHAP (*Shapley Additive Explanations*) [Lundberg and Lee 2017] é um *framework* agnóstico de modelo para explicação de predições em aprendizado de máquina. Sua abor-

dagem utiliza o valor de Shapley, oriundo da Teoria dos Jogos, que distribui os ganhos de um jogo entre seus jogadores dependendo da contribuição de cada um. Esse valor é obtido a partir da margem de contribuição média de cada jogador quando ele é adicionado em qualquer coalizão possível de jogadores. No contexto de explicação de predições de modelos de GNN, os elementos do grafo são os jogadores num jogo onde a predição final conta com a contribuição de cada um.

Levando em conta essas características, o SHAP é responsável por unificar outros métodos de explicação existentes sob uma nova definição: a classe de "métodos de atribuição de características aditivas". Dentro dessa classe, o *framework* demonstra teoricamente que os valores de Shapley são a única solução que satisfaz um conjunto de propriedades desejáveis específicas.

O SHAP constitui um método de explicação voltado especificamente para interpretações locais. Essa característica confere ao método uma abordagem mais direcionada e menos complexa em comparação às abordagens globais, que, por sua vez, ao visar compreender a influência das variáveis no modelo como um todo, apresentam alta complexidade. Afinal, a explicação global mais fiel é o próprio conjunto de parâmetros aprendidos, que, em modelos complexos, pode ser muito extenso e humanamente inviável de interpretar. Assim, o enfoque local adotado pelo SHAP favorece a geração de explicações mais claras e específicas.

Tratando-se de modelos complexos de predição, é interessante a identificação de um modelo de explicação mais simples, que pode ser qualquer aproximação interpretável do modelo original. Sendo f o modelo original e g o modelo de explicação responsável por aproximar a saída de f para uma instância específica, a interpretabilidade muitas vezes requer uma simplificação dos próprios dados de entrada. Essa simplificação é formalizada pelo uso de uma versão da entrada original x, representada por x', que pode ser mapeada de volta ao seu formato original por meio de uma função $x = h_x(x')$. De maneira semelhante, em tarefas de Processamento de Linguagem Natural (PLN), *word embeddings*, uma representação de palavras comumente utilizada, não é facilmente interpretável. Portanto, para fins de explicação, é mais interpretável a sua conversão para um formato mais simples e intuitivo, tal como *bag of words*. Dessa forma, o objetivo central dos métodos de explicação local é garantir que a saída do modelo simples $g(z')$, seja uma aproximação fiel da saída do modelo original $f(h_x(z'))$, ou seja, que $g(z') \approx f(h_x(z'))$ sempre que $z' \approx x'$.

No *framework* SHAP, o modelo explicativo é baseado no método de atribuição de características aditiva, uma função linear de valores binários, definido por:

$$g(z') = \Phi_0 + \sum_{i=1}^{M} \Phi_i z'_i$$

onde $z' \in \{0,1\}^M$ e indica a presença da característica (*feature*) na explicação. No contexto de GNNs, essa característica corresponde ao componente estrutural sob análise (nós, arestas ou subgrafos). M é o número de entradas simplificadas e $\Phi_i \in \Re$ é a contribuição da característica para a predição.

Assim, o método de atribuição de características aditivo atribui um valor Φ_i para cada característica, de modo que só haverá contribuição se a característica estiver pre-

sente na explicação. A soma de todas as contribuições aproxima-se de $f(x)$, a saída do modelo original. É notável que métodos de explicação proeminentes e desenvolvidos anteriormente, como LIME [Huang et al. 2020] e DeepLIFT [Shrikumar et al. 2017], se enquadram nessa definição, demonstrando a natureza unificadora do *framework* SHAP.

Além disso, esse modelo de atribuição de características aditivas possui três propriedades herdadas da Teoria dos Jogos, são elas:

1. **Local accuracy:** exige que o modelo de explicação $g(x')$ corresponda ao modelo original $f(x)$ quando $x = h_x(x')$ e $\phi_0 = f(h_x(0))$, onde Φ_0 representa a saída do modelo com todas as entradas "desligadas". Ou seja, o modelo de explicação deve reproduzir a saída do modelo original quando a entrada simplificada corresponder à entrada original.
2. **Missingness:** estabelece que atributos faltantes não devem impactar o resultado do modelo, ou seja, quando $x_i = 0$, não há influência atribuída;
3. **Consistency:** garante que, caso um modelo mude, de forma que a contribuição de algum dos atributos aumente ou permaneça igual, a influência dessa característica nunca deve diminuir. Isso pode ser matematicamente representado da seguinte forma: com $f(z') = f(h_x(z'))$, $z' \setminus i$ implicando $z'_i = 0$, e dois modelos sendo f e f', se $f'_x(z') - f'_x(z' \setminus i) \geq f_x(z') - f_x(z' \setminus i)$ para todas as entradas $z' \in \{0, 1\}^M$, então $\Phi_i(f', x) \geq \Phi_i(f, x)$.

A partir dessas três propriedades, é estabelecido que apenas um único modelo de explicação g consegue satisfazê-las. Esse modelo é conhecido como valores de Shapley e é dado pela seguinte equação:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!}[f_x(z') - f_x(z' \setminus i)]$$

onde $|z'|$ é o número de entradas não nulas em z', e $z' \subseteq x'$ representa todos os subconjuntos de entradas não nulas de x'.

A importância desse resultado reside no fato de que os valores de SHAP representam a única solução possível, dentro da família dos modelos de atribuição de características aditivas, que satisfaz simultaneamente as três propriedades fundamentais, algo que as metodologias unificadas pelo *framework* nem sempre garantem.

A computação exata dos valores de Shapley é computacionalmente custosa, o que levou ao desenvolvimento de modelos de aproximação para esses valores: o Kernel SHAP, uma abordagem geral e agnóstica de modelo que ajusta uma regressão linear local ponderada para encontrar os valores de Shapley; e métodos específicos ao modelo, como o Linear SHAP, que realiza a aproximação diretamente a partir dos coeficientes de modelos lineares, Low-Order SHAP, para modelos com poucas variáveis, Max SHAP, para modelos que envolvem cálculo do valor máximo entre as entradas, o Deep SHAP, para modelos de redes neurais profundas e o TreeShap para modelos de árvore de decisão [Lundberg et al. 2019]. Assim, é possível aplicar explicações baseadas em SHAP de forma mais prática e escalável, sem abrir mão da interpretabilidade do modelo.

4. Métodos de Explicação para GNNs Baseados no SHAP

Diante da complexidade apresentada pelo modelo da GNN, a interpretabilidade de seus resultados torna-se essencial para sua compreensão e confiabilidade. Enquanto mode-

los simples de aprendizado de máquina têm a capacidade de utilizar do próprio modelo para interpretações de resultado, abordagens mais complexas, como as GNNs, exigem um modelo de explicação simplificado. Nesse cenário, o *framework Shapley Additive Explanations* (SHAP) pode se apresentar como uma alternativa de explicação. A seção a seguir apresenta métodos de explicação aplicados a GNNs fundamentados nos valores de Shapley e com propriedades específicas, oriundas da Teoria dos Jogos. Serão analisadas, em ordem cronológica, as contribuições do GraphSVX [Duval and Malliaros 2021], SubgraphX [Yuan et al. 2021], EdgeSHAPer [Mastropietro et al. 2022], SAME [Ye et al. 2023] e GNNShap [Akkas and Azad 2024].

4.1. Explicações com GraphSVX

O GraphSVX [Duval and Malliaros 2021] surgiu como um dos primeiros métodos notáveis a aplicar o conceito de valor de Shapley às GNNs, apresentando explicações com embasamento teórico e inteligibilidade humana. Esse método é capaz de analisar a contribuição individual de um nó e suas respectivas características para uma predição específica, distinguindo-se por considerar, em conjunto, a estrutura do grafo nas suas explicações, ampliando o escopo da análise.

O algoritmo do GraphSVX segue o seguinte *pipeline*:

1. São criadas duas máscaras binárias a partir do algoritmo de amostragem SmarterSeparate, priorizando amostras com poucos ou quase todos os elementos incluídos. Para tamanhos intermediários, o algoritmo realiza uma amostra ponderada, garantindo que os nós e atributos apareçam com frequência equilibrada. Apenas uma pequena fração final das amostras (cerca de 10%) é utilizada de fato. Esse processo resulta em duas máscaras:
   - a primeira, uma máscara de nós ($M_N \in \{0,1\}^N$), que atua sobre a dimensão dos N nós do grafo.
   - a segunda, uma máscara de características ($M_F \in \{0,1\}^F$) que opera sobre a dimensão das F características do nó.

   Para neutralizar o valor de um nó específico, são apagadas suas conexões ao nó alvo, isolando-o, e, para neutralizar o valor de uma característica do nó, seu valor é substituído pelo valor médio dessa característica no *dataset*. Isso é feito porque as GNNs geralmente não lidam bem com valores ausentes.
2. As duas máscaras formam a coalizão $z = (M_F||M_N)$ e são utilizadas para gerar subgrafos $z' = (X', A')$ a partir do grafo original $g = (X, A)$, no qual $X \in \mathfrak{R}^{N\times F}$ é a matriz de características e $A \in \mathfrak{R}^{N\times N}$, a matriz de adjacências. Um tratamento para nós desconectados também é realizado: caso um nó distante seja incluído, porém os nós intermediários que o liguem ao nó alvo não, um algoritmo de caminho mínimo entre os dois nós é aplicado e os nós intermediários recebem características de valor médio, para que sua influência seja apenas estrutural.
3. Para medir a influência de cada coalizão, cada subgrafo z' é submetido ao modelo de predição original f para obter a predição $f(z')$. Essa predição reflete o comportamento do modelo quando apenas os elementos definidos pela máscara z estão ativos. Os resultados de todas as perturbações são compilados em um *dataset* $D = \{(z, f(z'))\}$.
4. Por fim, o gerador de explicações utiliza as máscaras e os subgrafos gerados para construir um modelo de explicação, mais especificamente uma Regressão Linear

Ponderada, em cima do *dataset* D para entender a importância de cada componente do grafo em sua predição. Os parâmetros aprendidos para cada componente são os valores de Shapley. A função de perda para treinar esse modelo é:

$$L_{f,\pi}(g) = \sum_{z \in Z} (g(z) - f(z'))^2 \pi_z$$

onde L é a função de perda a ser minimizada, g é o modelo de explicação, f é o modelo original, z são as amostras do *dataset* D e z' as amostras do gerador de grafos e π_z é um *kernel* de pesos atribuído a cada amostra, representando a importância do componente. No caso específico do trabalho, o *kernel* atribui um peso maior para amostras z que possuem dimensão pequena e grande, uma vez que se acredita que os efeitos individuais são mais fáceis de serem capturados a partir do efeito combinado nestes casos.

O GraphSVX, então, provê explicações considerando tanto os nós e suas características quanto a topologia do grafo.

4.2. Explicações com SubgraphX

O SubgraphX [Yuan et al. 2021] é um método aplicado em GNNs focado na exploração de subgrafos para explicação de predições do modelo. O estudo dos subgrafos é embasado na procura de explicações mais intuitivas, visto que estruturas do grafo são de grande importância para definir suas funcionalidades de maneira muito mais profunda do que a proporcionada pelo estudo de apenas nós e arestas, principalmente em áreas como bioquímica, ecologia, neurologia e engenharia. Para alcançar esse objetivo, o SubgraphX explora diferentes subgrafos de um grafo de entrada pelo algoritmo *Monte Carlo Tree Search* (MCTS), que é otimizado com o uso dos valores de Shapley.

O *Monte Carlo Tree Search* (MCTS) é um algoritmo de busca que utiliza a simulação de Monte Carlo para guiar a exploração de um espaço de busca. No contexto da explicabilidade de GNNs, o MCTS é empregado para explorar eficientemente diferentes subgrafos para um dado grafo de entrada. O algoritmo constrói uma árvore de busca na qual a raiz é associada ao grafo de entrada e os nós subsequentes correspondem a subgrafos conectados, obtidos por meio de ações de poda. Para direcionar a busca e reduzir o espaço de exploração, o MCTS registra estatísticas internas visando identificar o subgrafo com a pontuação mais alta.

Formalizando o problema, dado um modelo treinado de GNN $f(.)$ a partir de um grafo G, sendo y a classe predita, o objetivo do SubgraphX é encontrar o subgrafo conectado G^* que melhor explique essa predição, definido por:

$$G^* = argmax_{|G_i| \leq N_{min}} Score(f(.), G, G_i),$$

onde o conjunto $\{G_1, .., G_i, ..., G_n\}$ representa os subgrafos conectados de G, no qual n é o número total de subgrafos. $Score(f(.), G, G_i)$ é uma função de pontuação que analisa a importância de um subgrafo considerando o modelo $f(.)$ e o grafo G. N_{min} corresponde ao tamanho máximo dos subgrafos analisados. A exploração eficiente do espaço de busca é realizada pelo MCTS. A raiz, denotada pelo nó N_0, corresponde ao grafo de entrada G, os nós subsequentes, denotados por N_i são os subgrafos conectados e as arestas indicam

que o nó-filho é obtido a partir de uma ação de poda do nó-pai. Ao final do processo de busca, o subgrafo que obtém a maior pontuação é selecionado como a explicação final para a predição.

Para definir a importância de cada subgrafo (ou seja, a função $Score(f(.), G, G_i)$ usada como recompensa para guiar a busca MCTS), o SubgraphX utiliza o valor de Shapley. Contudo, o cálculo exato do valor de Shapley é computacionalmente inviável, exigindo duas aproximações principais para tornar a avaliação eficiente:

1. Primeiro, o método se baseia na observação de que, em GNNs com L camadas, a agregação de informações é restrita a uma vizinhança limitada de *L-hops* (saltos). Portanto, para calcular a importância de um subgrafo G_i, o SubgraphX considera que ele interage apenas com os nós dentro dessa vizinhança, e não com o grafo inteiro. Isso permite uma aproximação eficiente ao reduzir drasticamente o número de nós nas coalizões.
2. Mesmo com essa vizinhança restrita, testar todas as combinações de coalizões ainda é muito custoso. Por isso, o método utiliza a Amostragem de Monte-Carlo para estimar o valor de Shapley da seguinte forma:

$$\phi(G_i) = \frac{1}{T} \sum_{t=1}^{T} (f(S_i \cup \{G_i\}) - f(S_i)),$$

onde T é o número total de amostras de Monte-Carlo, e S_i é uma coalizão aleatória de nós sorteada daquela vizinhança restrita de L-hops. O termo $f(S_i \cup \{G_i\}) - f(S_i)$ mede a contribuição marginal média do subgrafo G_i para aquela amostra específica. Em resumo, está sendo calculada a contribuição marginal média do subgrafo G_i sobre T diferentes coalizões aleatórias.

Para o cálculo dessa contribuição marginal, o SubgraphX adota uma estratégia de *zero-padding* (preenchimento com zeros) nas características dos nós que não pertencem à coalizão (S_i) nem ao subgrafo (G_i), em vez de remover os nós. Isso é feito porque grafos são muito sensíveis a mudanças estruturais e essa abordagem preserva a topologia original.

4.3. Explicações com EdgeSHAPer

O EdgeSHAPer [Mastropietro et al. 2022] é outro método de explicação para modelos de GNN, projetado para interpretar predições específicas focando na importância das arestas de um grafo. Para isso, ele utiliza os valores de Shapley para quantificar a contribuição de cada conexão no resultado final. Essa abordagem centrada em arestas se destaca especialmente na análise de compostos químicos, onde sua principal vantagem é, também, a sua capacidade de identificar o conjunto mínimo de características pertinentes para uma predição específica do modelo. Apesar de ter sido concebido e avaliado no contexto de predições de compostos químicos, sua metodologia é generalizável.

Para contornar a alta complexidade do cálculo real dos valores de Shapley, o EdgeSHAPer utiliza uma estratégia de aproximação com a amostragem de Monte Carlo, onde as iterações estimam a contribuição de cada aresta j de um grafo G, sobre a previsão do modelo. O processo iterativo pode ser descrito nas seguintes etapas:

1. Para cada iteração, um grafo aleatório Z é gerado com o mesmo número de nós que G. Cada aresta e existente em G tem uma probabilidade P, análoga à densidade do grafo G, de existir em Z.
2. Uma permutação aleatória é criada para embaralhar a ordem das arestas. Essa permutação é aplicada tanto no conjunto de arestas reais de G, denominado E, quando ao conjunto de arestas do grafo Z, denominado E_Z.
3. A partir das listas ordenadas, dois novos conjuntos de arestas são construídos com o objetivo de isolar o impacto da aresta j:
    - A primeira, E_{+j}, é formado por arestas de E até a posição da aresta j e completando com arestas de E_Z nas posições restantes.
    - A segunda, E_{-j}, é idêntica à primeira, porém substitui a aresta j por sua contraparte vinda de Z.
4. Os dois conjuntos híbridos são submetidos ao modelo f e a diferença $f(E_{+j}) - f(E_{-j})$ resulta na contribuição marginal da aresta j para aquela iteração específica.

Para garantir a precisão, esse processo é repetido por um número M de passos de Monte Carlo e a média é calculada. Finalmente, após a execução do algoritmo para todas as arestas do grafo, o resultado é um *ranking* que ordena as arestas com base em seus valores de Shapley aproximados, identificando as arestas mais importantes para a predição final.

4.4. Explicações com SAME

O SAME (*Structure-Aware Shapley-Based Multipiece Explanation*)[Ye et al. 2023] é um método de explicação de GNN concebido para possibilitar a identificação da importância de características topológicas nas predições de um modelo. Modelos desenvolvidos anteriormente eram marjoritariamente caracterizados pela análise de uma única componente para explicação, como nós ou arestas, ignorando casos em que componentes conectados influenciam a predição significativamente. A principal contribuição do SAME é sua capacidade de gerar explicações compostas por múltiplas peças, conectadas ou não, fundamentadas teoricamente no valor de Shapley e cientes da estrutura do grafo.

O cálculo exato da importância de cada componente utilizando os valores de Shapley exige a avaliação da contribuição marginal de todas as combinações de nós possíveis no grafo, tornando esse processo computacionalmente caro. Para contornar essa problemática e introduzir sensibilidade à estrutura do grafo, o SAME utiliza uma adaptação nomeada de *k-hop* Shapley, que restringe a contribuição marginal apenas à vizinhança local num raio de k saltos. Para isso, é utilizada a seguinte fórmula:

$$I(f(\cdot), G^i_{ex}, G) = \sum_{p_i \subseteq P_{k-hop}} \frac{|p_i|!(|P_{k-hop}| - |p_i| - 1)!}{|P_{k-hop}|!}(f(p_i \cup G^i_{ex}) - f(p_i))$$

onde $I(f(\cdot), G^i_{ex}, G)$ representa a contribuição marginal da explicação G^i_{ex} para o modelo de predição $f(\cdot)$, no contexto do grafo de entrada G. A notação p_i refere-se a um subconjunto de nós vizinhos à explicação; P_{k-hop} é o conjunto total de jogadores, definido por $\{G^i_{ex}, v_{n+1}, v_{n+2}, ..., v_{n_i+k}\}$, englobando os nós da explicação G^i_{ex} e seus vizinhos até k saltos de distância. $f(p_i \cup G^i_{ex}) - f(p_i)$ mede o ganho na predição do modelo $f(\cdot)$ quando a explicação G^i_{ex} se junta a uma coalizão p_i.

O algoritmo do SAME, em sua primeira fase, utiliza a busca em árvore Monte Carlo (MCTS) no grafo para identificar subestruturas candidatas. O objetivo é encontrar um conjunto de componentes que possam apresentar grande importância para a predição. A importância de cada subestrutura é avaliada durante a busca, garantindo apenas componentes relevantes ao final da exploração. Após sua identificação, essas subestruturas passam por uma segunda busca MCTS, explorando combinações candidatas para a explicação final, que pode ser composta por uma ou mais das componentes previamente selecionadas. O objetivo é maximizar a pontuação de importância global. Assim, o SAME consegue construir explicações que refletem a influência conjunta de múltiplas partes do grafo. A utilização do *k-hop* Shapley como função de pontuação garante que essa busca seja não apenas justa, mas também ciente da estrutura e computacionalmente tratável.

4.5. Explicações com GNNShap

Entre os métodos de explicação aplicados a GNNs, o GNNShap [Akkas and Azad 2024] surgiu como uma resposta a limitações comuns de abordagens anteriores. Muitos dos métodos já existentes, como o GraphSVX [Duval and Malliaros 2021], utilizam amostras limitadas para estimar os valores de Shapley e tendem a considerar apenas coalizões de nós muito pequenas ou muito grandes, ignorando subconjuntos de tamanho intermediário, que poderiam conter informações úteis para as explicações. Além disso, são computacionalmente custosos, o que os torna inviáveis para grafos de tamanho moderado ou grande. O GNNShap busca superar essas limitações ao gerar explicações mais eficientes e com maior fidelidade na tarefa de classificação de nós por meio de arestas.

O algoritmo do GNNShap é dividido em quatro etapas:

1. **Poda das arestas redundantes:** as arestas que não carregam informação para o nó alvo, ou seja, não são parte de um caminho até o nó alvo, são podadas, reduzindo o custo computacional significativamente e sem afetar a qualidade da explicação.
2. **Amostragem de coalizões:** dentro do contexto de modelos de explicação com o valor de Shapley, a amostragem é uma alternativa para o uso de todas as coalizões possíveis, tarefa computacionalmente inviável. A seleção de quais coalizões participarão das amostras no GNNShap é aleatória, mas é implementada de forma específica como uma amostragem aleatória ponderada. Primeiro, o método utiliza o kernel de ponderação do SHAP para determinar quantas amostras devem ser coletadas de cada tamanho de coalizão existente. A fórmula desse kernel, que define o peso total (ρ) para todas as coalizões de um tamanho específico $|S|$, é:

$$\rho_{|S|} = \frac{n-1}{|S|(n-|S|)}$$

   onde n é o número total de arestas e $|S|$ é o tamanho da coalizão. Essa ponderação garante que todos os tamanhos de coalizão (pequenos, médios e grandes) sejam representados na amostragem final. Em seguida, o método seleciona aleatoriamente o número de coalizões determinado. Por fim, todas as coalizões de mesmo tamanho são geradas em paralelo na GPU, o que ajuda a capturar a topologia do grafo de forma mais representativa e eficiente.
3. **Predição do modelo:** antes do processo de predição de cada uma das amostras geradas, é realizada uma poda de coalizões onde o nó alvo está desconectado, ou

seja, não recebe informações de vizinhos relevantes. Outra otimização aplicada é o uso de *batching*, concatenando a matriz de adjacências e atributos dos nós em blocos. Essa organização permite a execução paralela das predições, otimizando significativamente o desempenho do modelo.

4. **Computação do valor de Shapley:** a aproximação dos valores de Shapley é computada seguindo a seguinte fórmula:

$$\phi = (M^T W M)^{-1} M^T W \hat{y}$$

Onde ϕ é o vetor resultante que contém os valores de importância aproximados, M é a matriz de máscara binária com dimensão $s \times n$ na qual s é o número de amostras de subgrafos e n, o número de componentes do grafo, W é uma matriz diagonal de pesos de dimensão $s \times s$, onde cada valor representa o peso associado à i-ésima amostra e $\hat{y}$ é o vetor de predições com n elementos. Essa equação corresponde à solução de uma regressão linear ponderada, ajustando ϕ de forma a minimizar o erro entre as predições do modelo e aquelas reconstruídas pela combinação linear das máscaras.

Assim, o GNNShap consegue contornar a problemática do alto custo computacional e a dificuldade em eliminar arestas irrelevantes para a predição utilizando-se da poda e paralelização dos processos.

5. Análise comparativa de Métodos de Explicação para GNNs e Discussão

A escolha dos métodos de explicabilidade para GNNs limitou-se à métodos com base teórica no *framework* SHAP, devido à sua fundamentação em axiomas que garantem uma atribuição de importância justa e consistente. Embora compartilhem essa base conceitual, a individualidade de cada método reside nas diferentes abordagens e estratégias para adaptar esses princípios ao domínio dos grafos de forma computacionalmente viável e respeitando a estrutura relacional dos grafos.

Essa busca por viabilidade é essencial pois, embora os Valores de Shapley sejam ainda o estado da arte para atribuições de importância, seu cálculo exato possui complexidade exponencial. Consequentemente, por necessidade prática, os métodos analisados baseiam-se em aproximações. Para isso, empregam técnicas como decomposição, perturbação e modelos substitutos para estimar a contribuição individual da componente do grafo escolhida.

A Tabela 1 apresenta uma tabela comparativa dos métodos de explicação para GNNs que utilizam ou se inspiram no *framework* SHAP. Nela, são detalhadas as seguintes características: *i*) a garantia dos Valores de Shapley refere-se ao nível de confiabilidade do cálculo desses valores; *ii*) o escopo do modelo indica a aplicabilidade do método em relação aos diferentes tipos de GNNs; *iii*) a técnica de explicação descreve a abordagem teórica central de cada modelo; *iv*) o método de perturbação é a estratégia geral usada para criar as entradas modificadas (amostras) avaliadas pelo modelo; *v*) a estratégia de amostragem detalha a lógica de seleção por trás dessas amostras; *vi*) a computação dos Valores de Shapley é a fórmula de cálculo utilizada para a aproximação; *vii*) as tarefas suportadas são os tipos de tarefas para as quais cada método pode fornecer explicações; *viii*) o componente para explicação é a unidade do grafo que recebe a pontuação de importância; *ix*) a representação dos componentes para perturbação é a ferramenta específica ou o formato

Tabela 1. Tabela comparativa dos métodos de explicação para GNNs que utilizam ou se inspiram no *framework* SHAP.

	GraphSVX	**SubgraphX**	**EdgeSHAPer**	**SAME**	**GNNShap**
Garantia dos Valores de Shapley	Aproximação	Aproximação	Aproximação	Aproximação	Aproximação
Escopo do modelo	Agnóstico ao modelo	Agnóstico ao modelo	Agnóstico ao modelo	Agnóstico ao modelo	Agnóstico ao modelo
Técnica de explicação	Decomposição, perturbação e modelo substituto	Decomposição e perturbação	Decomposição e perturbação	Perturbação	Decomposição, perturbação e modelo substituto
Método de perturbação	Amostragem de coalizões	Monte Carlo Tree Search (MCTS)	Amostragem de Monte Carlo	Monte Carlo Tree Search (MCTS)	Amostragem de coalizões
Estratégia de Amostragem	Ponderada para extremos	Busca guiada por recompensa (MCTS)	Aleatória baseada em permutação	Busca guiada por recompensa (MCTS)	Ponderada para todos os tamanhos
Computação dos Valores de Shapley	$(Z^TWZ)^{-1}Z^TW\hat{y}$	$\frac{1}{T}\sum_{t=1}^{T}(f(S_i \cup \{G_i\}) - f(S_i))$	$f(E_{+j}) - f(E_{-j})$	*K-hop* Shapley	$(M^TWM)^{-1}M^TW\hat{y}$
Tarefas suportadas	Classificação de nós e grafos	Classificação de nós e grafos	Classificação de grafos	Classificação de nós e grafos	Classificação de nós
Componente para explicação	Nós e atributos	Subgrafos conectados	Arestas	Subgrafos	Arestas
Representação dos componentes para perturbação	Máscara binária sobre nós e atributos	*Zero-padding* em atributos	Permutação e troca de arestas	*Zero-padding* em atributos	Máscara binária sobre arestas
Máscara dos nós	$M_N \in \{0,1\}^N$	N/A	N/A	N/A	N/A
Máscara dos atributos	$M_F \in \{0,1\}^F$	N/A	N/A	N/A	N/A
Máscara das arestas	N/A	N/A	Listas E_{+j} e E_{-j}	N/A	$M = k \times n$
Tamanho das coalizões no *dataset*	Pequenas e grandes	Todos os tamanhos	Todos os tamanhos	Todos os tamanhos	Todos os tamanhos

usado para aplicar essa perturbação no grafo; *x*) a máscara dos nós indica a representação da máscara de perturbação de nós, quando aplicável; *xi*) a máscara dos atributos indica a representação da máscara de perturbação de atributos, quando aplicável; *xii*) a máscara das arestas indica a representação da máscara ou lista de perturbação de arestas, quando aplicável; *xiii*) e o tamanho das coalizões no *dataset* descreve a cobertura dos tamanhos de subgrafos (coalizões) utilizados na amostragem.

A divergência mais significativa dos métodos reside, então, no elemento do grafo ao qual a importância é atribuída. A abordagem do GraphSVX é baseada na decomposição da predição ao nível dos nós e seus atributos, mesclando a contribuição da estrutura e seu conteúdo. Contrastando com essa visão, métodos como GNNShap e EdgeSHAPer focam nas arestas, reconhecendo que a propagação de informações nas GNNs ocorre, primariamente, por meio delas. Essa abordagem é valorizada em casos onde a relação é mais crucial que um nó isolado, como numa ligação química, por exemplo. Num nível de abstração maior, o SubgraphX busca pelo subgrafo conectado mais importante, partindo da ideia de que subestruturas inteiras são mais relevantes e inteligíveis. Por fim, o SAME expande essa mesma ideia ao fornecer explicações compostas por múltiplas peças estruturais, conectadas ou não, reconhecendo que a predição pode ser influenciada por diferentes partes do grafo.

Com a componente definida, é necessário lidar com a implementação dessas diferentes granularidades. Uma das estratégias de aproximação utilizadas é a amostragem por coalizões, utilizada pelo GraphSVX, GNNShap e EdgeSHAPer. Essa amostragem consiste na seleção de um subconjunto, chamado de coalizões, de todas as combinações possíveis de jogadores (características, nós ou arestas) para estimar de forma eficiente as contribuições marginais e, consequentemente, os Valores de Shapley. Abordagens ini-

ciais, como a do GraphSVX foram criticadas por focar em coalizões muito pequenas ou grandes, comprometendo a representabilidade. O GNNShap supera isso ao distribuir amostras eficientemente por todos os tamanhos de coalizão, aplicando otimizações e paralelização do processo. A segunda estratégia, utilizada pelo SAME e SubgraphX é a Monte Carlo Tree Search, permitindo a exploração do espaço de busca de forma a garantir que apenas subestruturas promissoras sejam avaliadas.

Com base nessa análise comparativa, argumenta-se que a divergência entre os métodos de explicação de GNNs reflete uma evolução conceitual no que constitui uma explicação útil. Há uma progressão nítida que se afasta de componentes granulares, como nós, atributos e arestas individuais, em direção a unidades semanticamente mais complexas, como subgrafos e subestruturas. Essa trajetória sugere um reconhecimento de que, para a inteligibilidade humana, a explicação de padrões estruturais pode ser mais valiosa que nós ou arestas isoladas, que por si só podem não carregar o contexto e complexibilidade total de uma predição.

Essa mudança na granularidade da explicação, no entanto, impacta diretamente o dilema entre fidelidade, inteligibilidade e eficiência computacional. Como o cálculo exato do SHAP é inviável, a escolha de focar em subgrafos em vez de arestas altera fundamentalmente o desafio da aproximação. Argumenta-se, portanto, que a escolha do método de explicação não deve ser universal. Métodos focados em arestas podem oferecer maior fidelidade em tarefas onde a importância da conexão é primordial. Em contraste, métodos focados em subgrafos são mais adequados para domínios onde a subestruturas compõem uma explicação inteligível. A decisão deve ser guiada pela natureza da tarefa e pelo tipo de entendimento que se deseja obter: focado em relações individuais ou focado em padrões estruturais.

6. Conclusão

A análise comparativa deste trabalho identificou que os métodos de explicação de GNNs alinhados ao SHAP, embora conceitualmente unificados, divergem, fundamentalmente, nas escolhas sobre qual componente do grafo utilizar para explicações e qual técnica de amostragem aplicar para contornar a complexidade computacional do cálculo dos valores de Shapley. A evolução dos métodos, partindo de componentes granulares para estruturas semanticamente mais ricas, sugere que a escolha da abordagem de explicação depende da natureza da tarefa e do tipo de entendimento desejado. Por exemplo, para a interpretação humana em domínios como a química, focar em arestas pode ser mais intuitivo, ao passo que em análises de redes sociais, a explicação por nós pode se sobressair. Já em tarefas onde padrões estruturais são centrais, como na identificação de *motifs* em bioinformática ou na detecção de círculos de fraude em redes financeiras a explicação por subgrafos pode oferecer uma interpretação mais valiosa.

Essas divergências técnicas introduzem um dilema fundamental entre eficiência computacional, fidelidade e inteligibilidade da explicação. O *framework* SHAP, fundamentado em valores da Teoria dos Jogos, oferece uma base teórica robusta nesse sentido e, aliado a estratégias de amostragem, apresenta uma base sólida para esse conflito.

Como trabalhos futuros, seria relevante a avaliação desses métodos em *benchmarks* controlados, comparando métodos focados em diferentes componentes para entender o impacto de sua escolha na prática e analisar o *trade-off* entre fidelidade e inteli-

gibilidade. Além disso, aproximações dos valores de Shapley computacionalmente mais eficientes e que preservem os axiomas do SHAP são de grande valor.

Referências

Akkas, S. and Azad, A. (2024). Gnnshap: Scalable and accurate gnn explanation using shapley values. In *Proceedings of the ACM Web Conference 2024*, WWW '24, page 827–838. ACM.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396.

Duval, A. and Malliaros, F. D. (2021). Graphsvx: Shapley value explanations for graph neural networks.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., and Yin, D. (2019). Graph neural networks for social recommendation.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558.

Huang, Q., Yamada, M., Tian, Y., Singh, D., Yin, D., and Chang, Y. (2020). Graphlime: Local interpretable model explanations for graph neural networks. *CoRR*, abs/2001.06216.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting model predictions.

Lundberg, S. M., Erion, G. G., and Lee, S.-I. (2019). Consistent individualized feature attribution for tree ensembles.

Markov, A. A. (2006). An example of statistical investigation of the text eugene onegin concerning the connection of samples in chains. *Science in Context*, 19(4):591–600.

Mastropietro, A., Pasculli, G., Feldmann, C., Rodríguez-Pérez, R., and Bajorath, J. (2022). Edgeshaper: Bond-centric shapley value-based explanation method for graph neural networks. *iScience*, 25(10):105043.

Peng, N., Poon, H., Quirk, C., Toutanova, K., and tau Yih, W. (2017). Cross-sentence n-ary relation extraction with graph lstms.

Réau, M., Renaud, N., Xue, L. C., and Bonvin, A. M. J. J. (2022). Deeprank-gnn: a graph neural network framework to learn patterns in protein–protein interfaces. *Bioinformatics*, 39(1):btac759.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through propagating activation differences. *CoRR*, abs/1704.02685.

Wu, Y., Lian, D., Xu, Y., Wu, L., and Chen, E. (2020). Graph convolutional networks with markov random field reasoning for social spammer detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):1054–1061.

Yang, L., Yang, G., Chen, X., Yang, Q., Yao, X., Bing, Z., Niu, Y., Huang, L., and Yang, L. (2021). Deep scoring neural network replacing the scoring function components to

improve the performance of structure-based molecular docking. *ACS Chemical Neuroscience*, 12(12):2133–2142. PMID: 34081851.

Ye, Z., Huang, R., Wu, Q., and Liu, Q. (2023). Same: Uncovering gnn black box with structure-aware shapley-based multipiece explanations. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 6442–6466. Curran Associates, Inc.

Yuan, H., Yu, H., Wang, J., Li, K., and Ji, S. (2021). On explainability of graph neural networks via subgraph explorations.

Zhou, D., Bousquet, O., Lal, T., Weston, J., and Schölkopf, B. (2003). Learning with local and global consistency. In Thrun, S., Saul, L., and Schölkopf, B., editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press.