Faithful and Accurate Self-Attention Attribution for Message Passing Neural Networks via the Computation Tree Viewpoint

Yong-Min Shin[1], **Siqing Li**[2], **Xin Cao**[2], **Won-Yong Shin**[1*]
[1]Yonsei University
[2]University of New South Wales
jordan3414@yonsei.ac.kr, {siqing.li, xin.cao}@unsw.edu.au, wy.shin@yonsei.ac.kr

Abstract

The self-attention mechanism has been adopted in various popular message passing neural networks (MPNNs), enabling the model to adaptively control the amount of information that flows along the edges of the underlying graph. Such attention-based MPNNs (Att-GNNs) have also been used as a baseline for multiple studies on explainable AI (XAI) since attention has steadily been seen as natural model interpretations, while being a viewpoint that has already been popularized in other domains (*e.g.,* natural language processing and computer vision). However, existing studies often use naïve calculations to derive attribution scores from attention, undermining the potential of attention as interpretations for Att-GNNs. In our study, we aim to fill the gap between the widespread usage of Att-GNNs and their potential *explainability* via attention. To this end, we propose GATT, edge attribution calculation method for self-attention MPNNs based on the *computation tree*, a rooted tree that reflects the computation process of the underlying model. Despite its simplicity, we empirically demonstrate the effectiveness of GATT in three aspects of model explanation: faithfulness, explanation accuracy, and case studies by using both synthetic and real-world benchmark datasets. In all cases, the results demonstrate that GATT greatly improves edge attribution scores, especially compared to the previous naïve approach.

Code — https://github.com/jordan7186/GAtt

1 Introduction

Background & motivation. In graph learning, graph neural networks (GNNs) (Wu et al. 2021) have been used as the *de facto* architecture, since they can effectively encode the graph structure along with node (or edge) features. Among various GNNs, several models have successfully incorporated the self-attention mechanism (Vaswani et al. 2017) into message passing neural networks (MPNNs) (Gilmer et al. 2017; Bronstein et al. 2021). Such (self-)attention-based MPNNs (dubbed **Att-GNNs**) have been one of the staple GNN architectures, and the self-attention mechanism of Att-GNNs themselves has been extensively analyzed in the literature (Knyazev, Taylor, and Amer 2019; Lee et al. 2019; Sun et al. 2023).[1] Furthermore, several studies have focused solely on analyzing GAT (Velickovic et al. 2018), the most representative Att-GNN model (Mustafa, Bojchevski, and Burkholz 2023; Fountoulakis et al. 2023).

Similarly as in other neural network models, GNNs are regarded as black-box models that lack interpretability, which has led to numerous studies developing explanation methods for GNNs (Li et al. 2022; Yuan et al. 2023). While such explanation methods have been widely developed, attention has also been frequently considered as a fundamental tool for GNN explanations (Ying et al. 2019; Luo et al. 2020; Sánchez-Lengeling et al. 2020). The choice of attention as a baseline is natural, as self-attention itself can be viewed as a direct way to provide model interpretations without any separate explanation method (Lee, Shin, and Kim 2017; Ghaeini, Fern, and Tadepalli 2018; Hao et al. 2021; Aflalo et al. 2022; Deiseroth et al. 2023). This viewpoint has already been extensively investigated in transformers, the most representative architecture with attention (Bahdanau, Cho, and Bengio 2015; Xu et al. 2015; Vig 2019; Dosovitskiy et al. 2021; Caron et al. 2021). There is even a significant body of research debating the validity of self-attention as explanations in natural language processing (NLP) (Jain and Wallace 2019; Wiegreffe and Pinter 2019; Bibal et al. 2022). However, there has been no such in-depth discussion from the domain of GNN explanations, mostly employing the layer-wise average of attention weights retrieved from a GAT model as explanations at best.

We argue that **such naïve usage of attention for interpretations largely undermines the potential of Att-GNNs as an explainable model**. In the case of transformers, a number of advanced attribution methods using attention have been proposed to calculate token attributions, and have been empirically proven that attention can be effectively used to decipher the underlying model (Abnar and Zuidema 2020; Chefer, Gur, and Wolf 2021a,b; Hao et al. 2021). Analogous to transformers, our study aims to formulate a post-processing method for the attention weights in Att-GNNs that is able to extract high-quality *edge attributions* (*i.e.*, to assign contributions of edges to the model) and capture the behavior of Att-GNNs more precisely. To the

*Corresponding Author.

[1]As such, we refer to the term 'GNN' as MPNN-style architectures unless explicitly stated otherwise.

Model	Naïve methods	Advanced methods
Transformers	*Raw attention* (Vig 2019)	(Abnar and Zuidema 2020) (Chefer, Gur, and Wolf 2021a) (Chefer, Gur, and Wolf 2021b) (Hao et al. 2021) etc.
Att-GNNs	*Layer-wise averaging* (Ying et al. 2019) (Luo et al. 2020)	**This work (GATT)**

Table 1: Overview of prior works on using attention as explanations. Despite various methods being developed for calculating token attributions for transformers, no corresponding method has yet been developed for calculating edge attributions for Att-GNNs.

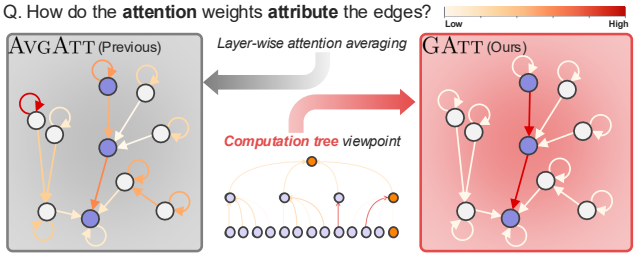


Figure 1: An visualization of our method (GATT, right) against the previous approach (AVGATT, left) on the Infection dataset, where the correct infection path is highlighted as the blue nodes.

best of our knowledge, we are the first to address this issue within the scope of general Att-GNNs, thus filling in the literature of explanations via attention (see the red part of Table 1).

Main contributions. In this study, we address the problem of developing an effective edge attribution method using attention weights in Att-GNNs. Our key insight is that **edge attributions with attention can be advanced by aligning with the feed-forward process of MPNNs**, *i.e.*, thinking in terms of the *computation tree* viewpoint, a rooted subtree that shows the local computation structure around a target node (see the middle part of Figure 1). Based on observing the computation tree, we assert that the edge attribution function should encompass two crucial principles: **P1)** proximity to the target node and **P2)** its position in the computation tree, thus aligning with the feed-forward process.

To this end, we introduce GATT, a simple yet effective solution to the edge attribution problem by integrating the *computation tree* of a given target node. Specifically, GATT adds attention weights in the underlying Att-GNN across the computation tree while adjusting their influence by employing targeted multiplication factors for attention weights guiding towards the target node. As an example, Figure 1 visualizes edge attribution scores from different edge attribution calculation methods using the same model. The attribution scores from GATT (see the right red box in Figure 1) show that the model places high emphasis on the correct infection path (highlighted as blue nodes). Such conclusion could not have been reached if we were to use simple layer-wise averaging (see the left box in Figure 1) as the tool for interpretations. To prove the effectiveness of GATT, we run extensive experiments by answering pivotal facets of interpretations—*faithfulness* and *explanation accuracy*—of Att-GNNs across diverse real-world and synthetic datasets. Despite the simplicity of GATT, empirical results demonstrate that the application of GATT to process attention weights within the underlying model produces substantively improved explanation capabilities, excelling in both faithfulness and explanation accuracy. We also perform an ablation study in which we introduce two variants of GATT, namely GATT_{SIM} and GATT_{AVG}, each of which corresponds to a removal of one critical design element (*i.e.*, **P1** or **P2**) of our method. Our analysis reveals clear deterioration of the quality of the edge attributions in all measures for both variants, which justifies the necessity of our two design elements. Finally, we remark that GATT is a straightforward calculation module (*i.e.*, does not involve any optimization/learning process), therefore brings the benefit of being hyperparameter-free and deterministic. In summary, we conclude that **Att-GNNs are indeed highly explainable** when adopting the proper interpretation, *i.e.,* **adjustment of attention weights** by **taking the viewpoint of the computation tree**. Note that graph transformers (Ying et al. 2021; Kreuzer et al. 2021; Chen et al. 2023) are beyond the scope of this study since transformers have already been analyzed and advanced by numerous studies (see Table 1). Our contributions are summarized as follows:

- **Key observations:** We make key observations and design principles that are crucial in edge attribution calculation by integrating the computation tree of the target node during its feed-forward process.
- **Novel methodology:** We propose GATT, a new method to calculate edge attributions from attention weights in Att-GNNs by integrating the computation tree of the given GNN model.
- **Extensive evaluations:** We extensively demonstrate that Att-GNNs are shown to be more faithful and accurate when using our proposed method compared to the simple alternative.

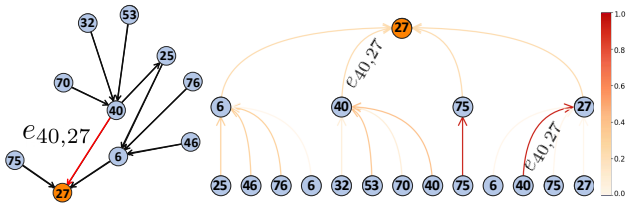
It should be noted that as long as Att-GNN architectures are employed, GATT is *model-agnostic* and standalone without any learning modules. We refer to (Shin et al. 2024) for a comprehensive review of related studies.

2 Edge Attribution Calculation in Att-GNNs

In this section, we first describe the notation used in the paper. Then, we formalize the problem of calculating edge attributions in Att-GNNs, and propose GATT, an approach to incorporate the computation tree into edge attributions.

2.1 Notations

Let us denote a given undirected graph as a tuple $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. We denote the edge connecting two nodes $v_i, v_j \in \mathcal{V}$ as



(a) 2-hop subgraph. (b) Computation tree.

Figure 2: A visualization for a 2-layer Att-GNN on target node 27 on the infection dataset. Figure 2a shows the local 2-hop subgraph with the edge $e_{40,27}$ marked as red. Figure 2b shows the computation tree in the Att-GNN, where the information flows from leaf nodes to node 27 at the root. The edges are colored by the attention weights from the model, while highlighting the two occurrences of edge $e_{40,27}$.

$e_{ij} \in \mathcal{E}$. We consider undirected graphs, *i.e.*, $e_{j,i} \in \mathcal{E}$ if $e_{i,j} \in \mathcal{E}$. The set of neighbors of node v_i is denoted as $\mathcal{N}_i$.

2.2 Problem Statement

We are given a graph $G = (\mathcal{V}, \mathcal{E})$, the Att-GNN model f with L layers, and a target node $v_i \in \mathcal{V}$ of interest. The attention weights calculated from f are denoted as $\mathcal{A} = \{\mathbf{A}(l)\}_{l=1}^{L}$, where $\mathbf{A}(l) \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ and $[\mathbf{A}(l)]_{j,i} = \alpha_{i,j}^{l}$ is the attention weight of edge $e_{i,j}$ in the l-th layer (with $l = 1$ being the input layer). The problem of edge attribution calculation is characterized by an edge attribution function $\Phi(v, \mathcal{A}, e_{i,j}) \triangleq \phi_{i,j}^{v}$ such that the *edge attribution score* $\phi_{i,j}^{v}$ accounts for the contribution of edge $e_{i,j}$ to the underlying model's calculation for node v (*i.e.,* faithfulness to f).

In our study, our objective is to design Φ using the *computation tree* in Att-GNNs alongside several observations and key design principles, which will be specified later. To design such a function Φ, we argue that the **computation tree of Att-GNNs** should be considered for the precise calculation of $\phi_{i,j}^{v}$, incorporating its several key properties. Note that, although most post hoc instance-level explanation methods for GNNs (Ying et al. 2019; Luo et al. 2020) also have a similar objective in terms of calculating $\phi_{i,j}^{v}$, they do not take advantage of attention weights $\mathcal{A}$. Additionally, although we mainly consider *node-level* tasks throughout the paper as a representative task, we also demonstrate that the idea of GATT can also be extended for *graph-level* tasks, which we refer to (Shin et al. 2024) for further experimental results.

2.3 From Attention to Attribution

We first visualize the computation tree in a Att-GNN, which will lead to several important observations to guide GATT, an edge attribution calculation method given the attention weights in the Att-GNN model.

Visualizing the Computation Tree. To provide an illustrative example, we train a 2-layer GAT model (Velickovic et al. 2018) with a single attention head on the synthetic infection benchmark dataset (Faber, Moghaddam, and Wattenhofer 2021). Figure 2a shows the 2-hop subgraph from target node 27, which contains all nodes and edges that the model involves from node 27's point of view. The computation tree in the GAT is commonly expressed as a *rooted subtree* (Sato, Yamada, and Kashima 2021), as shown in Figure 2b for node 27. In the figure, the information flows from leaf nodes at depth 2 to the root node 27 at depth 0, which exhibits an apparently different structure from that of the subgraph in Figure 2a. Note that the attention weights are calculated in each graph attention layer for each edge in $\mathcal{E}$.

Design Principles. We begin by making several observations from the computation tree:

(O1) Identical edges can appear multiple times in the computation tree. For example, edge $e_{40,27}$ in Figure 2a appears twice in Figure 2b.

(O2) Nodes do not appear uniformly in the computation tree. Specifically, nodes that are k-hops away from the target node do not exist in depth k' for $0 < k' < k$ (*e.g.,* node 70 appears only at depth 2 while node 40 appears three times).

(O3) The graph attention layer always includes self-loops during its feed-forward process.

Based on the above observations, we would like to state two design principles that are desirable when designing the edge attribution function Φ.

(P1) Proximity effect: Edges within closer proximity to the target node tend to highly impact the model's prediction compared with distant edges, since they **are likely to appear more frequently** in the computation tree.

(P2) Contribution adjustment: The contribution of an edge in the computation tree should be **adjusted** by its position (*i.e.*, other edges in the path towards the root).

By these standards, we revisit Figure 2b. We first see that edges close to the target node such as $e_{40,27}$ appear twice, whereas distant edges such as $e_{70,40}$ appear only once (**P1**). Moreover, we empirically show that this principle (**P1**) holds in most real-world datasets (which we refer to (Shin et al. 2024) for further details). Additionally, for the attention weights from the last graph attention layer (*i.e.,* edges connecting nodes at depth 1 to the root node), each edge tends to have roughly the value of 0.25 for the attention weights. In consequence, the information flowing from the first graph attention layer (*i.e.,* edges connecting leaf nodes to nodes at depth 1) will be diminished by roughly 0.25 as it reaches the root node (**P2**).

Proposed Method. To design the edge attribution function Φ, we start by formally defining the computation tree alongside the *flow* and the *attention flow*.

Definition 2.1 (Computation tree)**.** The computation tree for an L-layer Att-GNN in our study is defined as a rooted subtree of height L with the target node as the root node. For each node in the tree at depth d, the neighboring nodes and itself are at depth $d + 1$ with edges directed towards node v.

According to Definition 2.1, we define the concept of flows in the computation tree.

Definition 2.2 (Flow in a computation tree)**.** Given a computation tree as a rooted subtree of height L with the root (target) node v, we define a flow $\lambda^l_{i,j,v}$ as the list of edges that sequentially appear in a path of length l starting from a given edge $e_{i,j}$ within the computation tree and ending with some edge $e_{*,v}$.[2] We indicate the k-th position within the flow (*i.e.*, starting from the bottom of the computation tree) as $\lambda^l_{i,j,v}(k)$ for $k \in [1, L]$. We denote the set of all flows in the computation tree with node v at its root that starts from edge $e_{i,j}$ with length $m \in [1, L]$ as $\Lambda^m_v(e_{i,j})$.

From Definition 2.2, it follows that $\lambda^l_{i,j,v}(1) = e_{i,j}$ and $\lambda^l_{i,j,v}(l) = e_{*,v}$ for all flows in $\Lambda_v(e_{i,j})$.

Definition 2.3 (Attention flow in a computation tree)**.** Given a flow $\lambda^m_{v_0,v_1,w} = [e_{v_0,v_1}, \cdots, e_{*,w}]$ of length $m \leq L$ for an L-layer Att-GNN model, we define an attention flow $\alpha[\lambda^m_{v_0,v_1,w}]$ as the corresponding attention weights assigned to each edge by the associated graph attention layers:

$$\alpha[\lambda^m_{v_0,v_1,w}] = [\alpha^{L-m+1}_{v_0,v_1}, \cdots, \alpha^L_{*,w}]. \tag{1}$$

Then, it follows that $\alpha[\lambda^m_{v_0,v_1,w}](i) = \alpha^{L-m+i}_{v_{i-1},v_i}$.

Example 1. In Figure 2b, $\Lambda_{27}(e_{40,27})$ includes two flows, *i.e.,* $\lambda^1_{40,27,27} = [e_{40,27}]$ and $\lambda^2_{40,27,27} = [e_{40,27}, e_{27,27}]$, along with the corresponding attention flows $\alpha[\lambda^1_{40,27,27}] = [0.25]$ and $\alpha[\lambda^2_{40,27,27}] = [0.9, 0.25]$, respectively.

Finally, we are ready to present GATT.

Definition 2.4 (GATT)**.** Given a target node v, an edge $e_{i,j}$ of interest, the set of flows, $\Lambda_v(e_{i,j})$, and the attention flows for all flows in $\Lambda_v(e_{i,j})$, we define the edge attribution of $e_{i,j}$ in L-layer Att-GNN as

$$\phi^v_{i,j} = \sum_{m'=1}^{L} \sum_{\lambda^{m'}_{i,j,v} \in \Lambda^{m'}_v(e_{i,j})} C(\alpha[\lambda^{m'}_{i,j,v}])\alpha[\lambda^{m'}_{i,j,v}](1), \tag{2}$$

where $C(\alpha[\lambda^m_{i,j,v}]) = \prod_{2 \leq k \leq m} \alpha[\lambda^m_{i,j,v}](k)$ (or 1 if $m = 1$). Eq. (2) can be interpreted as follows. We first find all occurrences of the target edge $e_{i,j}$ in the computation tree, and then re-weight its corresponding attention score (*i.e.,* $\alpha[\lambda^m_{i,j,v}](1)$) by the product of all attention weights that appear *after* $e_{i,j}$ (*i.e.,* $\alpha[\lambda^m_{i,j,v}](k)$ for $k \geq 2$) in the flow before the summation over all relevant flows. Next, let us turn to addressing how our design principles (**P1**) and (**P2**) are met. (**P1**) holds as we *add* the contributions from each flow rather than taking the average, therefore the total number of occurrences of $e_{i,j}$ is directly expressed in the edge attribution. **(P2)** is fulfilled by the adjustment factor $C(\alpha[\lambda^m_{i,j,v}])$, since its value is dependent on the position of $\lambda^m_{i,j,v}(1)$. Essentially, $C(\alpha[\lambda^m_{i,j,v}])$ takes the chain of calculation from an edge to the target node into account. We provide an insightful example below.

[2]As stated in **(O1)**, nodes/edges are not unique in the computation tree. Nonetheless, we will use the node indices assigned from the original graph and avoid differentiating them in the computation tree as long as it does not cause any confusion.

Example 2. Let us recall $\lambda^2_{40,27,27} = [e_{40,27}, e_{27,27}]$ and its attention flow $\alpha[\lambda^2_{40,27,27}] = [0.9, 0.25]$ on node 27 from Example 1. At face value, the contribution of edge $e_{40,27}$ within the flow $\lambda^2_{40,27,27}$ should be 0.9. However, this is inappropriate since the information will eventually get muted significantly by $\alpha^2_{27,27} = 0.25$; thus, we need to consider the adjustment factor $C(\alpha[\lambda^2_{40,27,27}])$ before calculating the final edge attribution. From Definition 2.4, the edge attribution $\phi^{27}_{40,27}$ from the attention weights is calculated as $\phi^{27}_{40,27} = 1 \times 0.25 + 0.25 \times 0.9 = 0.475$.

Efficient Calculation of GATT. Although GATT is defined as Eq. (2), directly using this to compute the edge attribution $\phi^v_{i,j}$ is not desirable since it involves constructing the computation tree in the form of a rooted subtree for each node v, as well as computing over all relevant attention flows, resulting in high redundancy during computation and not being proper for batch computation. To overcome these computational challenges, as another contribution, we introduce a *matrix-based computation* method that is much preferred in practice. To this end, we first define

$$\mathbf{C}_L(k) = \begin{cases} \mathbf{I}, & \text{if } k = 0, \\ \mathbf{A}(L)\mathbf{A}(L-1)\cdots\mathbf{A}(L-k+1), & \text{otherwise.} \end{cases}$$

Then, we would like to establish the following proposition.

Proposition 2.5. *For a given set of attention weights $\mathcal{A} = \{\mathbf{A}(l)\}_{l=1}^L$ for an L-layer Att-GNN with $L \geq 1$,* GATT *in Definition 2.4 is equivalent to*

$$\phi^v_{i,j} = \sum_{m=1}^{L} [\mathbf{C}_L(L-m)]_{v,j}[\mathbf{A}(m)]_{j,i}. \tag{3}$$

We refer to (Shin et al. 2024) for the proof. Proposition 2.5 signifies that GATT sums the attention scores, weighted by the sum of the products of attention weights $[\mathbf{A}(m)]_{j,i}$ along the paths from node j to node v, over all graph attention layers. We also provide another GATT calculation method optimized for *batch calculations* in (Shin et al. 2024).

Complexity Analysis. We first analyze the *computational complexity* of GATT with matrix-based calculation. According to Eq. (3), the bottleneck for calculating $\phi^v_{i,j}$ is to acquire $\prod_{k=m+1}^L \mathbf{A}(k)$. However, this matrix can be pre-computed and does not require re-calculation after its initial acquirement. Since we only count the number of multiplications in the summation, the computational complexity is finally given by $O(L)$, which is extremely efficient. Next, according to Eq. (3), the *memory complexity* requires delving into $\mathbf{C}_L(L-m)$ and $\mathbf{A}(m)$. For an L-layer Att-GNN, while storing all attention weights in $\mathbf{A}(m)$ requires $O(L|\mathcal{E}|)$, $\mathbf{C}_L(L-m)$ requires at most $O(L||T^{L-1}||_0)$, where T denotes the adjacency matrix and $||\cdot||_0$ is the 0-norm. In conclusion, the total memory complexity is bounded by $O(L||T^{L-1}||_0 + L|\mathcal{E}|)$. In addition to the above theoretical findings, we empirically provide runtime evaluations, which demonstrate that GATT is reasonably fast and scalable, achieving up to **58.05 times faster** runtime against PG-Explainer (Luo et al. 2020) when calculating edge attributions for 10,000 nodes, which we refer to (Shin et al. 2024) for detailed experimental results.

Dataset		2-layer GAT/GATv2			3-layer GAT/GATv2		
		GATT	AVGATT	Random	GATT	AVGATT	Random
Cora	Δ_{PC}	**0.8468/0.1040**	0.1764/0.0121	-0.0056/-0.0036	**0.8642/0.1696**	0.0967/0.0168	0.0045/0.0045
	Δ_{NE}	**0.7112/0.0930**	0.1526/0.0100	-0.0076/0.0019	**0.7690/0.1664**	0.0859/0.0186	0.0040/0.0037
	Δ_{P}	**0.9755/0.9623**	0.7251/0.6226	0.4389/0.4891	**0.9875/0.9966**	0.7075/0.8897	0.5235/0.6107
Citeseer	Δ_{PC}	**0.8516/0.0658**	0.3096/0.0180	0.0012/-0.0043	**0.8711/0.0432**	0.2110/0.0107	-0.0073/-0.0034
	Δ_{NE}	**0.7653/0.0700**	0.2780/0.0186	0.0021/0.0019	**0.8291/0.0551**	0.2006/0.0140	0.0015/0.0025
	Δ_{P}	**0.9846/0.9771**	0.9213/0.9510	0.3695/0.4258	**0.9920/0.9961**	0.8979/0.9692	0.4039/0.7569
Pubmed	Δ_{PC}	**0.8812/0.0631**	0.1648/0.0126	-0.0064/0.0021	**0.8489/0.0367**	0.0592/0.0023	0.0015/-0.0016
	Δ_{NE}	**0.8201/0.0915**	0.1477/0.0169	-0.0068/0.0078	**0.8612/0.0484**	0.0600/0.0028	0.0009/-0.0015
	Δ_{P}	**0.9915/0.9972**	0.8834/0.9361	0.3974/0.1327	**0.9993/0.9996**	0.8932/0.9153	0.5172/0.5242
Arxiv	Δ_{PC}	**0.7790/0.0546**	0.0794/-0.0593	0.0007/0.0028	**0.7721/0.0508**	0.0465/-0.0252	-0.0004/-0.0003
	Δ_{NE}	**0.8287/0.0164**	0.0804/-0.0390	0.0016/-0.0067	**0.8282**/-0.0012	0.0478/-0.0216	-0.0017/**0.0000**
	Δ_{P}	**0.9908/0.8995**	0.8470/0.2560	0.4962/0.5107	**0.9985/0.9366**	0.8331/0.3934	0.5004/0.5034
Cornell	Δ_{PC}	**0.8089/0.2660**	0.3391/0.0209	-0.0284/0.0421	**0.7173/0.0899**	0.3065/-0.0512	-0.0273/-0.0129
	Δ_{NE}	**0.7820/0.1526**	0.3199/-0.0488	-0.0231/0.0235	**0.7160/0.0520**	0.3491/-0.0294	-0.0060/-0.0017
	Δ_{P}	**0.9532/0.8372**	0.7416/0.5130	0.5074/0.5660	**0.9270/0.6406**	0.6907/0.3969	0.4787/0.4953
Texas	Δ_{PC}	**0.7818/0.0801**	0.3676/-0.0406	-0.0762/0.0025	**0.6866/0.1504**	0.2443/0.0486	0.0414/0.0040
	Δ_{NE}	**0.7977**/0.1443	0.3809/**0.1478**	-0.0659/0.0145	**0.6132/0.0896**	0.1645/0.0579	0.0202/0.0149
	Δ_{P}	**0.8726/0.7299**	0.6803/0.3669	0.4733/0.5198	**0.9197/0.8195**	0.7072/0.5565	0.5562/0.5426
Wisconsin	Δ_{PC}	**0.6898/0.1751**	0.2649/0.0556	0.0596/0.0120	**0.7616**/0.0323	0.3034/0.0337	-0.0059/**0.0407**
	Δ_{NE}	**0.6421/0.1554**	0.2340/0.0636	0.0414/0.0157	**0.7409**/0.0243	0.2762/**0.0574**	-0.0010/0.0400
	Δ_{P}	**0.8985/0.8501**	0.7067/0.6060	0.5427/0.5006	**0.8982/0.7582**	0.6906/0.3980	0.5119/0.5333

Table 2: Experimental results on the faithfulness measure for GATT, AVGATT, and random attribution for GAT/GATv2 on 7 real-world datasets. Results for 2/3-layer GAT/GATv2s are shown for each case (the best performer is highlighted as **bold**).

3 Can *Attention* Interpret Att-GNNs?

In this section, we carry out empirical studies to validate the effectiveness of GATT on interpreting two representative Att-GNN models: **GAT** (Velickovic et al. 2018) and **GATv2** (Brody, Alon, and Yahav 2022), with a single-attention head. Despite only a subset of all experimental results being presented due to page limitations, we have also demonstrated that GATT can be generally applied to other Att-GNNs by showing the results for another model, **SuperGAT** (Kim and Oh 2021). Additionally, we have found that the trend in performance for **multi-head** attention is consistent with the case for single-head attention. Finally, we have shown that the **regularization** during training has negligible effects on GATT. We refer to (Shin et al. 2024) for the details on the additional results.

3.1 Is *Attention* Faithful to the GNN?

We focus primarily on one of the most important properties in evaluating the performance of an explanation method: *faithfulness*, which measures how closely the attribution reflects the inner workings of the underlying model (Jacovi and Goldberg 2020; Chrysostomou and Aletras 2021; Liu et al. 2022; Li et al. 2022). Measuring the faithfulness involves 1) manipulating the input according to the attribution scores of interest and 2) observing the change in the model's response. We specify our experiment settings below.

Datasets. In our experiments, we use seven citation datasets. Specifically, we use four *homophilic* datasets, including Cora, Citeseer, Pubmed (Yang, Cohen, and Salakhutdinov 2016), and one *large-scale* dataset, Arxiv (Hu et al. 2020), and three *heterophilic* datasets, including Cornell, Texas, and Wisconsin (Pei et al. 2020). We refer to (Shin et al. 2024) for detailed descriptions including the dataset statistics.

Baseline Methods. Since the analysis of edge attribution from attention in Att-GNNs has not been studied previously, we present our own baseline approaches. We first compare the proposed GATT against another attention-based explanation method (Ying et al. 2019; Luo et al. 2020; Sánchez-Lengeling et al. 2020), named as AVGATT, which attributes each edge as the average of the attention weights over different layers and attention heads. We additionally include random attribution as another baseline ('Random'), by randomly assigning scores in $[0, 1]$ to each edge.

Attention Reduction. It is generally known that removing $e_{i,j}$ from the graph to measure its effect may cause the out-of-distribution problem (Hooker et al. 2019; Hase, Xie, and Bansal 2021), a common pitfall for perturbation-based approaches. To mitigate this, we opt mask the attention coefficients (*i.e.*, attention weights before softmax) corresponding to edge $e_{i,j}$ with zeros in the computation tree, which reduces the effect of $e_{i,j}$ without removal. Moreover, we do not mask the attention weights after softmax, which cannot occur in a normal feed-forward process of Att-GNNs since the attention distribution is not properly normalized. In other words, we only mask attention coefficients from one edge at a time, which is compared with the original response of the Att-GNN model (Tomsett et al. 2020).

Model	Dataset	GATT	AVGATT	SA	GB	IG	GNNEx	PGEx	GM	FDnX	Random
GAT	BA-Shapes	0.9591	0.7977	0.9563	0.6231	0.6231	0.8916	0.8289	0.5316	**0.9917**	0.4975
	Infection	**0.9976**	0.8786	0.8237	0.8949	0.9472	0.9272	0.7173	0.6859	0.6574	0.4811
GATv2	BA-Shapes	0.9617	0.7876	0.9626	0.5260	0.5232	0.9318	0.5000	0.5123	**0.9923**	0.4976
	Infection	**0.8628**	0.4719	0.7711	0.7250	0.7849	0.7611	0.8178	0.5355	0.5059	0.5002

Table 3: Experimental results on the explanation accuracy for the synthetic datasets using 3-layer GAT/GATv2s, measured in terms of the AUROC. The results for directly using attention weights as explanation are colored as red. The best and runner-up performers are marked as **bold** and underline, respectively, for each dataset and model.

Measurement. Denoting the output probability vector of Att-GNN for node v as $\mathbf{p}_v$ and the output probability vector after the attention reduction for $e_{i,j}$ as $\mathbf{p}_{v\setminus e_{i,j}}$, we measure the model's behavior from three points of view: 1) *decline in prediction confidence* Δ_{PC} (Guo et al. 2017) defined as the decrease of the probability for the predicted label (*i.e.,* $\Delta_{\text{PC}} = \mathbf{p}_v[k] - \mathbf{p}_{v\setminus e_{i,j}}[k]$, where $k = \arg\max_k \mathbf{p}_v[k]$), 2) *change in negative entropy* Δ_{NE} (Moon et al. 2020) defined as the increase of 'smoothness' of the probability vector (*i.e.,* $\Delta_{\text{NE}} = -\sum \mathbf{p}_{v\setminus e_{i,j}} \log \mathbf{p}_{v\setminus e_{i,j}} + \sum \mathbf{p}_v \log \mathbf{p}_v$), which also reflects the model's confidence, and 3) *change in prediction* Δ_{P} (Tomsett et al. 2020), which observes whether $k \neq k'$, where $\arg\max_k \mathbf{p}_v[k]$ and $\arg\max_{k'} \mathbf{p}_{v\setminus e_{i,j}}[k']$, where $\mathbf{p}_v[k]$ is the k-th entry of $\mathbf{p}_v$.

Quantitative Analysis of Faithfulness. We investigate the relationship between the model's output difference from attention reduction following edge attribution scores and the edge attribution scores themselves. In each dataset, we randomly select 100 nodes as target nodes v and calculate the values of GATT for all edges (i, j) that affect the target node (*i.e.*, $\phi^v_{i,j}$). We also perform attention reduction for the same edges (i, j) and measure Δ_{PC}, Δ_{NE}, and Δ_{P} to observe the correlation between GATT values. Specifically, we adopt the Pearson correlation for Δ_{PC} and Δ_{NE}. For Δ_{P}, we use the area under receiver operating characteristic (AUROC), basically measuring the quality of attribution scores as a predictor of whether the prediction of the target node will change after attention reduction.

Table 2 summarizes the experimental results with respect to the faithfulness on the seven real-world datasets, using pre-trained 2-layer and 3-layer GAT/GATv2s with a single attention head for each dataset. The results strongly indicate that GATT substantially increases the faithfulness of edge attributions of the GAT/GATv2s models, producing a more reliable attribution score compared to AVGATT and random attribution. Although AVGATT shows modest performance in Δ_{P}, it performs poorly in terms of changes in confidence (*i.e.,* Δ_{PC} and Δ_{NE}), sometimes performing worse than random attribution. This is because AVGATT does not account for the proximity effect and contribution adjustment and rather naïvely averages the attention weights over different layers and attention heads with no context of the computation tree. We refer to (Shin et al. 2024) for the detailed results on the consistent trend on multi-head attention, as well as additional results via *visualizations* by plotting histograms for Δ_{P}, which indicates that GATT successfully assesses whether the model prediction changes after attention reduction.

3.2 Does *Attention* Reveal Accurate Graph Explanations?

We evaluate the edge attributions of GATs in comparison with ground truth explanations. Since only the synthetic datasets are equipped with proper ground truth explanations, we only use these datasets during evaluations.

Datasets. We use the BA-shapes and Infection synthetic benchmark datasets. *BA-shapes* (Ying et al. 2019) attaches 80 house-shaped motifs to a base graph made from the Barabási-Albert model with 300 nodes, where the edges included in the motif are set as the ground truth explanations. *Infection benchmark* (Faber, Moghaddam, and Wattenhofer 2021) generates a backbone graph from the Erdös-Rényi model; then, a small portion of the nodes are assigned as 'infected', and the ground truth explanation is the path from an infected node to the target node. We expect that edge attributions should highlight such ground truth explanations for GATs with sufficient performance.[3]

Baseline Methods. In our experiments, we mainly compare among attention-based edge attribution calculation methods (*i.e.,* GATT and AVGATT) including Random attribution. Additionally, we consider seven popular *post-hoc* explanation methods: Saliency (SA) (Simonyan, Vedaldi, and Zisserman 2014), Guided Backpropagation (GB) (Springenberg et al. 2015), Integrated Gradient (IG) (Sundararajan, Taly, and Yan 2017), GNNExplainer (GNNEx) (Ying et al. 2019), PGExplainer (PGEx) (Luo et al. 2020), GraphMask (GM) (Schlichtkrull, Cao, and Titov 2021), and FastDnX (FDnX) (Pereira et al. 2023). We emphasize that post-hoc explanation methods are treated as a complementary tool of inherent explanations, thus belonging to a **different category** (Du, Liu, and Hu 2020). However, we include them for a more comprehensive comparison.

Experimental Results. Table 3 summarizes the results on the explanation accuracy for two synthetic datasets with ground truth explanations. As in prior studies (Ying et al. 2019; Luo et al. 2020), we treat evaluation as a binary classification of edges, aiming to predict whether each edge belongs to ground truth explanations by using the attribution scores as probability values. In this context, we adopt the AUROC as our metric. For both datasets, we observe that

[3]We refer to (Shin et al. 2024) for further descriptions and statistics of datasets.

Method	GATT	GATT$_{\text{SIM}}$	GATT$_{\text{AVG}}$	AVGATT
(**P1**) Proximity effect	✔	✔	✗	✗
(**P2**) Contribution adjustment	✔	✗	✔	✗

Table 4: Properties of different edge attribution methods.

Dataset	Model	GATT	GATT$_{\text{SIM}}$	GATT$_{\text{AVG}}$	AVGATT
Cora	2-layer	**0.8477**	0.7708	0.8109	0.1768
	3-layer	**0.8624**	0.6392	0.6900	0.0966
Citeseer	2-layer	**0.8516**	0.8058	0.4761	0.3096
	3-layer	**0.8711**	0.6671	0.8202	0.2110
Pubmed	2-layer	**0.8812**	0.7683	0.5915	0.1648
	3-layer	**0.8489**	0.4197	0.8302	0.0592

Table 5: Performance comparison among different edge attribution calculation methods for GATs.

GATT is much superior to AVGATT. Even compared to the representative post-hoc explanation methods, GATT shows a surprisingly competitive performance. For Infection, GATT shows the best performance, and while GATT places second and third for BA-Shapes, it still achieves over 0.95 AUROC scores. This indicates that the attention weights can inherently capture the GAT/GATv2s' behavior as long as the attribution calculation is provided by GATT.

3.3 Ablation & Case Study

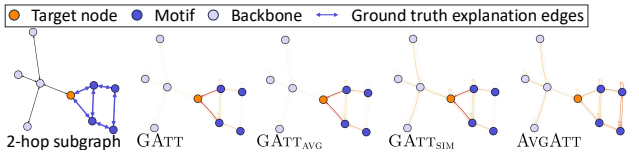
Ablation Study. GATT in Definition 2.4 is developed in the sense of satisfying the two design principles (*i.e.*, proximity effect (**P1**) and contribution adjustment (**P2**)). We now perform an ablation study to validate the effectiveness of each design element using the GAT model. To this end, we devise two variants GATT$_{\text{SIM}}$ and GATT$_{\text{AVG}}$ by simply adding all attention weights uniformly corresponding to the target edge in the computation tree and replacing the weighted summation in Eq. (2) with averaging to remove the effects of the proximity effect, respectively. More specifically, GATT$_{\text{SIM}}$ and GATT$_{\text{AVG}}$ are defined as

$$\sum_{m'=1}^{L} \sum_{\lambda^{m'}_{i,j,v} \in \Lambda^{m'}_{v}(e_{i,j})} \alpha[\lambda^{m}_{i,j,v}](1), \text{ and} \tag{4}$$

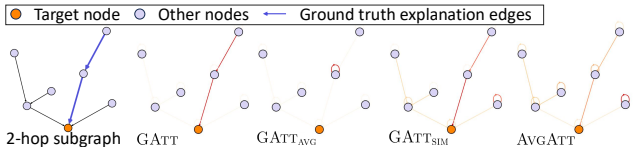
$$\frac{1}{|\Lambda_v(e_{i,j})|} \sum_{m'=1}^{L} \sum_{\lambda^{m'}_{i,j,v} \in \Lambda^{m'}_{v}(e_{i,j})} C(\alpha[\lambda^{m}_{i,j,v}])\alpha[\lambda^{m}_{i,j,v}](1), \tag{5}$$

respectively. The properties of different edge attribution calculation methods are summarized in Table 4.

We compare the performance among GATT, GATT$_{\text{SIM}}$, GATT$_{\text{AVG}}$, and AVGATT by running experiments with respect to the faithfulness on the Cora, Citeseer, and Pubmed datasets using GATs. Table 5 summarizes the results of ablation by reporting the Pearson's coefficient values for Δ_{PC}. We observe that GATT consistently outperforms both GATT$_{\text{SIM}}$ and GATT$_{\text{AVG}}$ for all cases. In particular, we observe the performance degradation of GATT$_{\text{SIM}}$ is generally more severe for 3-layer GATs. This is because the effects



(a) Node 679 on BA-Shapes.



(b) Node 2 on Infection.

Figure 3: Case study on the BA-Shapes and Infection datasets for a 2-layer GAT.

of the contribution adjustment (**P2**) and the cardinality of $\Lambda_v(e_{i,j})$ are more significant in a 3-layer GAT since the length of each attention flow is longer and the number of flows to consider is much higher compared to the case of 2-layer GATs.

Case Study. We conduct case studies on the BA-Shapes and Infection datasets for a 2-layer GAT, while visualizing how different methods behave. In Figure 3, each of two cases shows a randomly selected target node (marked as orange) and the edges in ground truth explanations (blue edges). We aim to observe how much the attribution scores from GATT, GATT$_{\text{AVG}}$ and GATT$_{\text{SIM}}$ focus on the ground truth explanation edges. Indeed, for both datasets, GATT focuses primarily on the edges in ground truth explanations, while the attribution scores from GATT$_{\text{SIM}}$ and AVGATT tend to be more spread throughout the entire 2-hop local graph. This indicates that the attention weights in the GAT indeed recognize the ground truth explanations under GATT calculations. In the case of GATT$_{\text{AVG}}$, the attribution patterns are not much different from GATT in BA-Shapes. However, GATT$_{\text{AVG}}$ in the Infection dataset attributes its attribution scores to a single self-loop edge that does not belong to the ground truth explanations, failing to provide adequate explanations. Interestingly, on BA-Shapes, GATT tends to strongly emphasize edges that are closer in proximity even within the house-shaped motifs, which coincides with the pitfall addressed in (Faber, Moghaddam, and Wattenhofer 2021). Further extensive case studies including *more target nodes* and *3-layer* models exhibit a similar tendency to Figure 3, which we refer to (Shin et al. 2024).

4 Conclusion and Future work

In this study, we have investigated the largely underexplored problem of interpreting Att-GNNs. Although Att-GNNs were not considered as a candidate for inherently explainable models, our empirical evaluations have demonstrated affirmative results when our proposed method, GATT, built upon the computation tree, can be used to effectively calculate edge attribution scores. Although GATT is generally applicable, this work does not include a systematic analysis on how different designs of attention weights will interact with GATT, which we leave for future work.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1A2C3004345, No. RS-2023-00220762).

References

Abnar, S.; and Zuidema, W. H. 2020. Quantifying Attention Flow in Transformers. In *ACL*, 4190–4197. Online.

Aflalo, E.; Du, M.; Tseng, S.; Liu, Y.; Wu, C.; Duan, N.; and Lal, V. 2022. VL-InterpreT: An Interactive Visualization Tool for Interpreting Vision-Language Transformers. In *CVPR*. New Orleans, LA.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*. San Diego, CA.

Bibal, A.; Cardon, R.; Alfter, D.; Wilkens, R.; Wang, X.; François, T.; and Watrin, P. 2022. Is Attention Explanation? An Introduction to the Debate. In *ACL*, 3889–3900. Dublin, Ireland.

Brody, S.; Alon, U.; and Yahav, E. 2022. How Attentive are Graph Attention Networks? In *ICLR*. Virtual Event.

Bronstein, M. M.; Bruna, J.; Cohen, T.; and Velickovic, P. 2021. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges. arXiv:2104.13478.

Caron, M.; Touvron, H.; Misra, I.; Jégou, H.; Mairal, J.; Bojanowski, P.; and Joulin, A. 2021. Emerging Properties in Self-Supervised Vision Transformers. In *ICCV*, 9630–9640. Montreal, Canada.

Chefer, H.; Gur, S.; and Wolf, L. 2021a. Generic Attention-model Explainability for Interpreting Bi-Modal and Encoder-Decoder Transformers. In *ICCV*, 387–396. Montreal, Canada.

Chefer, H.; Gur, S.; and Wolf, L. 2021b. Transformer Interpretability Beyond Attention Visualization. In *CVPR*, 782–791. virtual.

Chen, J.; Gao, K.; Li, G.; and He, K. 2023. NAGphormer: A Tokenized Graph Transformer for Node Classification in Large Graphs. In *ICLR*.

Chrysostomou, G.; and Aletras, N. 2021. Improving the Faithfulness of Attention-based Explanations with Task-specific Information for Text Classification. In *ACL/IJCNLP*, 477–488. Virtual Event.

Deiseroth, B.; Deb, M.; Weinbach, S.; Brack, M.; Schramowski, P.; and Kersting, K. 2023. ATMAN: Understanding Transformer Predictions Through Memory Efficient Attention Manipulation. In *NeurIPS*. New Orleans, LA.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*. Virtual event.

Du, M.; Liu, N.; and Hu, X. 2020. Techniques for interpretable machine learning. *Commun. ACM*, 63(1): 68–77.

Faber, L.; Moghaddam, A. K.; and Wattenhofer, R. 2021. When Comparing to Ground Truth is Wrong: On Evaluating GNN Explanation Methods. In *KDD*, 332–341. Virtual Event.

Fountoulakis, K.; Levi, A.; Yang, S.; Baranwal, A.; and Jagannath, A. 2023. Graph Attention Retrospective. *J. Mach. Learn. Res.*, 24: 246:1–246:52.

Ghaeini, R.; Fern, X. Z.; and Tadepalli, P. 2018. Interpreting Recurrent and Attention-Based Neural Models: a Case Study on Natural Language Inference. In *EMNLP*, 4952–4957. Brussels, Belgium.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural Message Passing for Quantum Chemistry. In *ICML*, 1263–1272. Sydney, NSW.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On Calibration of Modern Neural Networks. In *ICML*, 1321–1330. Sydney, Australia.

Hao, Y.; Dong, L.; Wei, F.; and Xu, K. 2021. Self-Attention Attribution: Interpreting Information Interactions Inside Transformer. In *AAAI*, 12963–12971. Virtual event.

Hase, P.; Xie, H.; and Bansal, M. 2021. The Out-of-Distribution Problem in Explainability and Search Methods for Feature Importance Explanations. In *NeurIPS*, 3650–3666. Virtual event.

Hooker, S.; Erhan, D.; Kindermans, P.; and Kim, B. 2019. A Benchmark for Interpretability Methods in Deep Neural Networks. In *NeurIPS*, 9734–9745. Vancouver, Canada.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open Graph Benchmark: Datasets for Machine Learning on Graphs. In *NeurIPS*. Virtual event.

Jacovi, A.; and Goldberg, Y. 2020. Towards Faithfully Interpretable NLP Systems: How Should We Define and Evaluate Faithfulness? In *ACL*, 4198–4205. Online.

Jain, S.; and Wallace, B. C. 2019. Attention is not Explanation. In *NAACL-HLT*, 3543–3556. Minneapolis, MN.

Kim, D.; and Oh, A. 2021. How to Find Your Friendly Neighborhood: Graph Attention Design with Self-Supervision. In *ICLR*. Virtual Event.

Knyazev, B.; Taylor, G. W.; and Amer, M. R. 2019. Understanding Attention and Generalization in Graph Neural Networks. In *NeruIPS*. Vancouver, Canada.

Kreuzer, D.; Beaini, D.; Hamilton, W. L.; Létourneau, V.; and Tossou, P. 2021. Rethinking Graph Transformers with Spectral Attention. In *NeurIPS*, 21618–21629. Virtual event.

Lee, J.; Shin, J.; and Kim, J. 2017. Interactive Visualization and Manipulation of Attention-based Neural Machine Translation. In *EMNLP*, 121–126. Copenhagen, Denmark.

Lee, J. B.; Rossi, R. A.; Kim, S.; Ahmed, N. K.; and Koh, E. 2019. Attention Models in Graphs: A Survey. *ACM Trans. Knowl. Discov. Data*, 13(6): 62:1–62:25.

Li, P.; Yang, Y.; Pagnucco, M.; and Song, Y. 2022. Explainability in Graph Neural Networks: An Experimental Survey. arXiv:2203.09258.

Liu, Y.; Li, H.; Guo, Y.; Kong, C.; Li, J.; and Wang, S. 2022. Rethinking Attention-Model Explainability through Faithfulness Violation Test. In *ICML*, 13807–13824. Baltimore, MD.

Luo, D.; Cheng, W.; Xu, D.; Yu, W.; Zong, B.; Chen, H.; and Zhang, X. 2020. Parameterized Explainer for Graph Neural Network. In *NeurIPS*. Virtual event.

Moon, J.; Kim, J.; Shin, Y.; and Hwang, S. 2020. Confidence-Aware Learning for Deep Neural Networks. In *ICML*, 7034–7044. Vienna, Austria.

Mustafa, N.; Bojchevski, A.; and Burkholz, R. 2023. Are GATs Out of Balance? In *NeurIPS*. New Orleans, LA.

Pei, H.; Wei, B.; Chang, K. C.; Lei, Y.; and Yang, B. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *ICLR*. Addis Ababa, Ethiopia.

Pereira, T. A.; Nascimento, E.; Resck, L. E.; Mesquita, D.; and Souza, A. H. 2023. Distill n' Explain: explaining graph neural networks using simple surrogates. In *AISTATS*. Palau de Congressos, Spain.

Sánchez-Lengeling, B.; Wei, J. N.; Lee, B. K.; Reif, E.; Wang, P.; Qian, W. W.; McCloskey, K.; Colwell, L. J.; and Wiltschko, A. B. 2020. Evaluating Attribution for Graph Neural Networks. In *NeurIPS*. virtual.

Sato, R.; Yamada, M.; and Kashima, H. 2021. Random Features Strengthen Graph Neural Networks. In *SDM*, 333–341. Virtual event.

Schlichtkrull, M. S.; Cao, N. D.; and Titov, I. 2021. Interpreting Graph Neural Networks for NLP With Differentiable Edge Masking. In *ICLR*. Virtual Event.

Shin, Y.; Li, S.; Cao, X.; and Shin, W. 2024. Faithful and Accurate Self-Attention Attribution for Message Passing Neural Networks via the Computation Tree Viewpoint. *CoRR*, abs/2406.04612.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In *ICLR*. Banff, Canada.

Springenberg, J. T.; Dosovitskiy, A.; Brox, T.; and Riedmiller, M. A. 2015. Striving for Simplicity: The All Convolutional Net. In *ICLR*. San Diego, CA.

Sun, C.; Li, C.; Lin, X.; Zheng, T.; Meng, F.; Rui, X.; and Wang, Z. 2023. Attention-based graph neural networks: a survey. *Artif. Intell. Rev.*, 56(S2): 2263–2310.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. In *ICML*. Sydney, Australia.

Tomsett, R.; Harborne, D.; Chakraborty, S.; Gurram, P.; and Preece2, A. 2020. Sanity Checks for Saliency Metrics. In *AAAI*. New York, NY.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *NIPS*, 5998–6008. Long Beach, CA.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *ICLR*. Vancouver, Canada.

Vig, J. 2019. A Multiscale Visualization of Attention in the Transformer Model. In *ACL*, 37–42. Florence, Italy.

Wiegreffe, S.; and Pinter, Y. 2019. Attention is not not Explanation. In *EMNLP-IJCNLP*, 11–20. Hong Kong, China.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. S. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Networks Learn. Syst.*, 32(1): 4–24.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A. C.; Salakhutdinov, R.; Zemel, R. S.; and Bengio, Y. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*, 2048–2057. Lille, France.

Yang, Z.; Cohen, W. W.; and Salakhutdinov, R. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *ICML*. New York City, NY.

Ying, C.; Cai, T.; Luo, S.; Zheng, S.; Ke, G.; He, D.; Shen, Y.; and Liu, T. 2021. Do Transformers Really Perform Badly for Graph Representation? In *NeurIPS*, 28877–28888. virtual.

Ying, Z.; Bourgeois, D.; You, J.; Zitnik, M.; and Leskovec, J. 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. In *NeurIPS*, 9240–9251. Vancouver, Canada.

Yuan, H.; Yu, H.; Gui, S.; and Ji, S. 2023. Explainability in Graph Neural Networks: A Taxonomic Survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(5): 5782–5799.