On Explainability of Graph Neural Networks via Subgraph Explorations

Hao Yuan [1]   Haiyang Yu [1]   Jie Wang [2]   Kang Li [3]   Shuiwang Ji [1]

Abstract

We consider the problem of explaining the predictions of graph neural networks (GNNs), which otherwise are considered as black boxes. Existing methods invariably focus on explaining the importance of graph nodes or edges but ignore the substructures of graphs, which are more intuitive and human-intelligible. In this work, we propose a novel method, known as SubgraphX, to explain GNNs by identifying important subgraphs. Given a trained GNN model and an input graph, our SubgraphX explains its predictions by efficiently exploring different subgraphs with Monte Carlo tree search. To make the tree search more effective, we propose to use Shapley values as a measure of subgraph importance, which can also capture the interactions among different subgraphs. To expedite computations, we propose efficient approximation schemes to compute Shapley values for graph data. Our work represents the first attempt to explain GNNs via identifying subgraphs explicitly and directly. Experimental results show that our SubgraphX achieves significantly improved explanations, while keeping computations at a reasonable level.

1. Introduction

Graph neural networks have drawn significant attention recently due to their promising performance on various graph tasks, including graph classification, node classification, link prediction, and graph generation. Different techniques have been proposed to improve the performance of deep graph models, such as graph convolution (Kipf & Welling, 2017; Gilmer et al., 2017; Gao et al., 2018; Wang et al., 2020; Yuan & Ji, 2021), graph attention (Veličković et al.,

2018; Wang et al., 2019), and graph pooling (Yuan & Ji, 2020; Gao & Ji, 2019; Zhang et al., 2018). However, these models are still treated as black boxes, and their predictions lack explanations. Without understanding and reasoning the relationships behind the predictions, these models cannot be understood and fully trusted, which prevents their applications in critical areas. This raises the need of investigating the explainability of deep graph models.

Recently, extensive efforts have been made to study explanation techniques for deep models on images and text (Simonyan et al., 2013; Yuan et al., 2019; Smilkov et al., 2017; Yuan et al., 2020a; Yang et al., 2019; Du et al., 2018). These methods can explain both general network behaviors and input-specific predictions via different strategies. However, the explainability of GNNs is still less explored. Unlike images and texts, graphs are not grid-like data and contain important structural information. Thus, methods for images and texts cannot be applied directly. While several recent studies have developed GNN explanation methods, such as GNNExplainer (Ying et al., 2019), PGExplainer (Luo et al., 2020), and PGM-Explainer (Vu & Thai, 2020), they invariably focus on explainability at node, edge, or node feature levels, and only consider subgraphs indirectly via regularization terms. We argue that subgraph-level explanations are more intuitive and useful, since subgraphs can be simple building blocks of complex graphs and are highly related to the functionalities of graphs (Alon, 2007; Milo et al., 2002).

In this work, we propose the SubgraphX, a novel GNN explanation method that can identify important subgraphs to explain GNN predictions. Specifically, we propose to employ the Monte Carlo tree search algorithm (Silver et al., 2017) to efficiently explore different subgraphs for a given input graph. Since the information aggregation procedures in GNNs can be interpreted as interactions among different graph structures, we propose to employ Shapley values (Kuhn & Tucker, 1953) to measure the importance of subgraphs by capturing such interactions. Furthermore, we propose efficient approximation schemes to Shapley values by considering interactions only within the information aggregation range. Altogether, our work represents the first attempt to explain GNNs via identifying subgraphs explicitly. We conduct both qualitative and quantitative experiments to evaluate the effectiveness and efficiency of our SubgraphX.

[1] Department of Computer Science & Engineering, Texas A&M University, TX, USA [2] Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China [3] West China Biomedical Big Data Center, West China Hospital, Chengdu, China. Correspondence to: Shuiwang Ji <sji@tamu.edu>.

Experimental results show that our proposed SubgraphX can provide better explanations for a variety of GNN models. In addition, our method has a reasonable computational cost given its superior performance.

2. Related Work

2.1. Graph Neural Networks

Graph neural networks have demonstrated their effectiveness on different graph tasks. Several approaches are proposed to learn representations for nodes and graphs, such as GCNs (Kipf & Welling, 2017), GATs (Veličković et al., 2018), and GINs (Xu et al., 2019), etc. These methods generally follow an information aggregation scheme that the features of a target node are obtained by aggregating and combining the features from its neighboring nodes. Here we use GCNs as an example to illustrate such information aggregation procedures. Formally, a graph $\mathcal{G}$ with m nodes can be represented by an adjacency matrix $A \in \{0, 1\}^{m \times m}$ and a feature matrix $X \in \mathbb{R}^{m \times d}$ assuming that each node is associated with a d-dimensional feature vector. Then the aggregation operation in GCNs can be mathematically written as $X_{i+1} = \sigma(D^{-\frac{1}{2}} \hat{A} D^{-\frac{1}{2}} X_i W_i)$, where X_i denotes the output feature matrix of $i-$th GCN layer and X_0 is set to $X_0 = X$. The node features are transformed from $X_i \in \mathbb{R}^{m \times c_i}$ to $X_{i+1} \in \mathbb{R}^{m \times c_{i+1}}$. Note that $\hat{A} = A + I$ is employed to add self-loops and D is a diagonal node degree matrix to perform normalization on $\hat{A}$. In addition, $W_i \in \mathbb{R}^{c_i \times c_{i+1}}$ is a learnable weight matrix to perform linear transformations on features and $\sigma(\cdot)$ is the non-linear activation function.

2.2. Explainability in Graph Neural Networks

Even though explaining GNNs is crucial to understand and trust deep graph models, the explainability of GNNs is still less studied, compared with the image and text domains. Recently, several methods are proposed specifically to explain deep graph models. These methods mainly focus on explaining GNNs by identifying important nodes, edges, node features. However, none of them can provide input-dependent subgraph-level explanations, which is important for understanding graph models. Following a recent survey work (Yuan et al., 2020c), we categorize these methods into several classes; those are, gradients/features-based methods, decomposition methods, surrogate methods, generation-based methods, and perturbation-based methods.

First, several methods employ gradient values or feature values to study the importance of input graph nodes, edges, or node features (Baldassarre & Azizpour, 2019; Pope et al., 2019). These methods generally extend existing image explanation techniques to the graph domain, such as SA (Zeiler & Fergus, 2014), CAM (Zhou et al., 2016), and Guided

BP (Springenberg et al., 2015). While these methods are simple and efficient, they cannot incorporate the special properties of graph data. Meanwhile, decomposition methods, such as LRP (Schwarzenberg et al., 2019), Excitation BP (Pope et al., 2019), and GNN-LRP (Schnake et al., 2020), explain GNNs by decomposing the original model predictions into several terms and associating these terms with graph nodes or edges. These methods generally follow a backpropagation manner to decompose predictions layer by layer until input space. In addition, existing methods (Huang et al., 2020; Vu & Thai, 2020) employ a simple and interpretable model as the surrogate method to capture local relationships of deep graph models around the input data. Then the explanations of the surrogate method are treated as the explanations of the original predictions. Furthermore, recent work XGNN (Yuan et al., 2020b) proposes to study general and high-level explanations of GNNs by generating graph patterns to maximize a certain prediction.

In addition, a popular direction to explain GNNs is known as the perturbation-based method. It monitors the changes in the predictions by perturbing different input features and identifies the features affecting predictions the most. For example, GNNExplainer (Ying et al., 2019) optimizes soft masks for edges and node features to maximize the mutual information between the original predictions and new predictions. Then the optimized masks can identify important edges and features. Meanwhile, PGExplainer (Luo et al., 2020) learns a parameterized model to predict whether an edge is important, which is trained using all edges in the dataset. It employs the reparameterization trick (Jang et al., 2016) to obtain approximated discrete masks instead of soft masks. In addition, GraphMask (Schlichtkrull et al., 2021a) follows a similar idea as PGExplainer that train a classifier to predict if an edge can be dropped without affecting model predictions. However, it studies the edges in every GNN layer while PGExplainer only focuses on the input space.

3. The Proposed SubgraphX

While most current methods for GNN explanations are invariably based on directly identifying important nodes or edges, we argue that directly studying important subgraphs is more natural and may lead to better explainability. In this work, we propose a novel approach, known as SubgraphX, to explain GNNs by exploring and identifying important subgraphs.

3.1. From Node and Edge to Subgraph Explanations

Unlike images and texts, graph data contain important structural information, which is highly related to the properties of graphs. For example, network motifs, which can be considered as graph substructures, are simple building blocks of complex networks and may determine the functionalities of
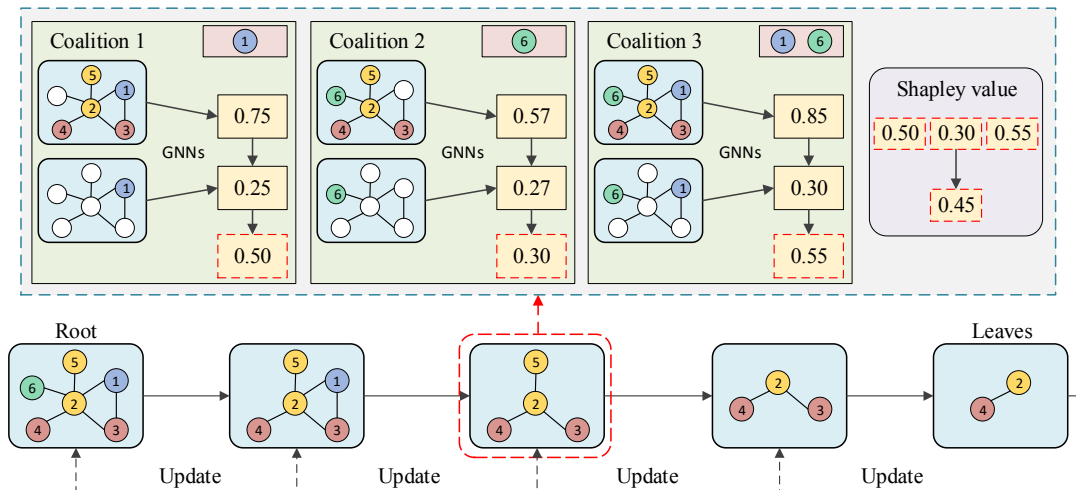
Figure 1. An illustration of our proposed SubgraphX. The bottom shows one selected path from the root to leaves in the search tree, which corresponds to one iteration of MCTS. For each node, its subgraph is evaluated by computing the Shapley value via Monte-Carlo sampling. In this example, we show the computation of Shapley value for the middle node (shown in red dashed box) where three coalitions are sampled to compute the marginal contributions. Note that nodes that are not selected are ignored for simplicity.

graphs in many domains, such as biochemistry, ecology, neurobiology, and engineering (Alon, 2007; Milo et al., 2002; Shen-Orr et al., 2002; Alon, 2019). Hence, investigating graph substructures is a crucial step towards the reverse engineering and understanding of the underlying mechanisms of GNNs. In addition, subgraphs are more intuitive and human-intelligible (Yuan et al., 2020c).

While different methods are proposed to explain GNNs, none of them can directly provide subgraph-level explanations for individual input examples. The XGNN can obtain graph patterns to explain GNNs but its explanations are not input-dependent and less precise. The other methods, such as GNNExplainer and PGExplainer, may obtain subgraph-level explanations by combining nodes or edges to form subgraphs in a post-processing manner. However, the important nodes or edges in their explanations are not guaranteed to be connected. Meanwhile, since GNNs are very complex, node/edge importance cannot be directly converted to subgraph importance. Furthermore, these methods ignore the interactions among different nodes and edges, which may contain important information. Hence, in this work, we propose a novel method, known as SubgraphX, to directly study the subgraphs to provide explanations. The explanations of our SubgraphX are connected subgraphs, which are more human-intelligible. In addition, by incorporating Shapley values, our method can capture the interactions among different graph structures when providing explanations.

3.2. Explaining GNNs with Subgraphs

We first present a formal problem formulation. Let $f(\cdot)$ denote the trained GNNs to be explained. Without loss

of generality, we introduce our proposed SubgraphX by considering $f(\cdot)$ as a graph classification model. Given an input graph $\mathcal{G}$, its predicted class is represented as y. The goal of our explanation task is to find the most important subgraph for the prediction y. Since disconnected nodes are hard to understand, we only consider connected subgraphs to enable the explanations to be more human-intelligible. Then the set of connected subgraphs of $\mathcal{G}$ is denoted as $\{\mathcal{G}_1, \cdots, \mathcal{G}_i, \cdots, \mathcal{G}_n\}$ where n is the number of different connected subgraphs in $\mathcal{G}$. The explanation of prediction y for input graph $\mathcal{G}$ can then be defined as

$$\mathcal{G}^* = \underset{|\mathcal{G}_i| \leq N_{\min}}{\operatorname{argmax}} \operatorname{Score}(f(\cdot), \mathcal{G}, \mathcal{G}_i), \qquad (1)$$

where $\operatorname{Score}(\cdot, \cdot, \cdot)$ is a scoring function for evaluating the importance of a subgraph given the trained GNNs and the input graph. We use $N_{\min}$ as an upper bound on the size of subgraphs so that the obtained explanations are succinct enough. A straightforward way to obtain $\mathcal{G}^*$ is to enumerate all possible $\mathcal{G}_i$ and select the most important one as the explanation. However, such a brute-force method is intractable when the graph is complex and large-scale. Hence, in this work, we propose to incorporate search algorithms to explore subgraphs efficiently. Specifically, we propose to employ Monte Carlo Tree Search (MCTS) (Silver et al., 2017; Jin et al., 2020) as the search algorithm. In addition, since the information aggregation procedures in GNNs can be understood as interactions between different graph structures, we propose to employ the Shapley value (Kuhn & Tucker, 1953) as the scoring function to measure the importance of different subgraphs by considering such interactions. We illustrate our proposed SubgraphX in Figure 1. After search-

ing, the subgraph with the highest score is considered as the explanation of the prediction y for input graph $\mathcal{G}$. Note that our proposed SubgraphX can be easily extended to use other search algorithms and scoring functions.

3.3. Subgraph Exploration via MCTS

In our proposed SubgraphX, we employ the MCTS as the search algorithm to guide our subgraph explorations. We build a search tree in which the root is associated with the input graph and each of other nodes corresponds to a connected subgraph. Each edge in our search tree denotes that the graph associated with a child node can be obtained by performing node-pruning from the graph associated with its parent node. Formally, we define a node in this search tree as $\mathcal{N}_i$, and $\mathcal{N}_0$ denotes the root node. The edges in the search tree represent the pruning actions a. Note that each node may have many pruning actions, and these actions can be defined based on the dataset at hand or domain knowledge. Then the MCTS algorithm records the statistics of visiting counts and rewards to guide the exploration and reduce the search space. Specifically, for the node and pruning action pair $(\mathcal{N}_i, a_j)$, we assume that the subgraph $\mathcal{G}_j$ is obtained by action a_j from $\mathcal{G}_i$. Then the MCTS algorithm records four variables for $(\mathcal{N}_i, a_j)$, which are defined as:

- $C(\mathcal{N}_i, a_j)$ denotes the number of counts for selecting action a_j for node $\mathcal{N}_i$.
- $W(\mathcal{N}_i, a_j)$ is the total reward for all $(\mathcal{N}_i, a_j)$ visits.
- $Q(\mathcal{N}_i, a_j) = W(\mathcal{N}_i, a_j)/C(\mathcal{N}_i, a_j)$ and denotes the averaged reward for multiple visits.
- $R(\mathcal{N}_i, a_j)$ is the immediate reward for selecting a_j on $\mathcal{N}_i$, which is used to measure the importance of subgraph $\mathcal{G}_j$. We propose to use $R(\mathcal{N}_i, a_j) = \text{Score}(f(\cdot), \mathcal{G}, \mathcal{G}_j)$.

In each iteration, the MCTS selects a path starting from the root $\mathcal{N}_0$ to a leaf node $\mathcal{N}_\ell$. Note that the leaf nodes can be defined based on the numbers of nodes in subgraphs such that $|\mathcal{N}_\ell| \leq N_{\min}$. Formally, the action selection criteria of node $\mathcal{N}_i$ are defined as

$$a^* = \underset{a_j}{\arg\max}\, Q(\mathcal{N}_i, a_j) + U(\mathcal{N}_i, a_j), \quad (2)$$

$$U(\mathcal{N}_i, a_j) = \lambda R(\mathcal{N}_i, a_j)\frac{\sqrt{\sum_k C(\mathcal{N}_i, a_k)}}{1 + C(\mathcal{N}_i, a_j)}, \quad (3)$$

where λ is a hyperparameter to control the trade-off between exploration and exploitation. In addition, $\sum_k C(\mathcal{N}_i, a_k)$ denotes the total visiting counts for all possible actions of node $\mathcal{N}_i$. Then the subgraph in the leaf node $\mathcal{N}_\ell$ is evaluated and the importance score is denoted as $\text{Score}(f(\cdot), \mathcal{G}, \mathcal{G}_\ell)$. Finally, all node and action pairs selected in this path are updated as

$$C(\mathcal{N}_i, a_j) = C(\mathcal{N}_i, a_j) + 1, \quad (4)$$
$$W(\mathcal{N}_i, a_j) = W(\mathcal{N}_i, a_j) + \text{Score}(f(\cdot), \mathcal{G}, \mathcal{G}_\ell). \quad (5)$$

After searching for several iterations, we select the subgraph with the highest score from the leaves as the explanation. Note that in early iterations, the MCTS tends to select child nodes with low visit counts in order to explore different possible pruning actions. In later iterations, the MCTS tends to select child nodes that yield higher rewards, *i.e.*, more important subgraphs.

3.4. A Game-Theoretical Scoring Function

In our proposed SubgraphX, both the MCTS rewards and the explanation selection are highly depending on the scoring function $\text{Score}(\cdot, \cdot, \cdot)$. It is crucial to properly measure the importance of different subgraphs. One possible solution is to directly feed the subgraphs to the trained GNNs $f(\cdot)$ and use the predicted scores as the importance scores. However, it cannot capture the interactions between different graph structures, thus affecting the explanation results. Hence, in this work, we propose to adopt the Shapley values (Kuhn & Tucker, 1953; Lundberg & Lee, 2017; Chen et al., 2018a) as the scoring function. The Shapley value is a solution concept from the cooperative game theory for fairly assigning a total game gain to different game players. To apply it to graph model explanation tasks, we use the GNN prediction as the game gain and different graph structures as players.

Formally, given the input graph $\mathcal{G}$ with m nodes and the trained GNN $f(\cdot)$, we study the Shapley value for a target subgraph $\mathcal{G}_i$ with k nodes. Let $V = \{v_1, \cdots, v_i, \cdots, v_m\}$ denote all nodes in $\mathcal{G}$ and we assume that the nodes in $\mathcal{G}_i$ are $\{v_1, \cdots, v_k\}$ while the other nodes $\{v_{k+1}, \cdots, v_m\}$ belong to $\mathcal{G} \setminus \mathcal{G}_i$. Then the set of players is defined as $P = \{\mathcal{G}_i, v_{k+1}, \cdots, v_m\}$, where we consider the whole subgraph $\mathcal{G}_i$ as one player. Finally, the Shapley value of the player $\mathcal{G}_i$ can be computed as

$$\phi(\mathcal{G}_i) = \sum_{S \subseteq P\setminus\{\mathcal{G}_i\}} \frac{|S|!\,(|P| - |S| - 1)!}{|P|!} m(S, G_i), \quad (6)$$

$$m(S, \mathcal{G}_i) = f(S \cup \{\mathcal{G}_i\}) - f(S), \quad (7)$$

where S is the possible coalition set of players. Note that $m(S, \mathcal{G}_i)$ represents the marginalized contribution of player $\mathcal{G}_i$ given the coalition set S. It can be computed by the difference of predictions between incorporating $\mathcal{G}_i$ with and without the coalition set S. The obtained Shapley value $\phi(\mathcal{G}_i)$ considers all different coalitions to capture the interactions. It is the only solution that satisfies four desirable axioms, including efficiency, symmetry, linearity, and dummy axiom (Lundberg & Lee, 2017), which can guarantee the correctness and fairness of the explanations. However, computing Shapley values using Eqs. (6) and (7) is time-consuming as it enumerates all possible coalitions, especially for large-scale and complex graphs. Hence, in this work, we propose to incorporate the GNN architecture information $f(\cdot)$ to efficiently approximate Shapley values.

Algorithm 1 The algorithm of our proposed SubgraphX.

Input: GNN model $f(\cdot)$, input graph $\mathcal{G}$, MCTS iteration number M, the leaf threshold node number $N_{\min}$, $h(\mathcal{N}_i)$ denotes the associated subgraph of tree node $\mathcal{N}_i$.

Initialization: for each $(\mathcal{N}_i, a_j)$ pair , initialize its C, W, Q, and R variables as 0. The root of search tree is $\mathcal{N}_0$ associated with graph $\mathcal{G}$. The leaf set is set to $S_\ell = \{\}$.

for $i = 1$ **to** M **do**
  $curNode = \mathcal{N}_0$, $curPath = [\mathcal{N}_0]$
  while $h(curNode)$ has more node than $N_{\min}$ **do**
    for all possible pruning actions of $h(curNode)$ **do**
      Obtain child node $\mathcal{N}_j$ and its subgraph $\mathcal{G}_j$.
      Compute $R(curNode, a_j) = \text{Score}(f(\cdot), \mathcal{G}, \mathcal{G}_j)$
      with Algorithm 2.
    end for
    Select the child $\mathcal{N}_{next}$ following Eq.(2, 3).
    $curNode = \mathcal{N}_{next}$, $curPath = curPath + \mathcal{N}_{next}$.
  end while
  $S_\ell = S_\ell \cup \{curNode\}$
  Update nodes in $curPath$ following Eq.(4, 5).
end for
Select subgraph with the highest score from S_ℓ.

Algorithm 2 The algorithm of subgraph Shapley value.

Input: GNN model $f(\cdot)$ with L layers, input graph $\mathcal{G}$ with nodes $V = \{v_1, \ldots, v_m\}$, subgraph $\mathcal{G}_i$ with k nodes $\{v_1, \ldots, v_k\}$, Monte-Carlo sampling steps T.

Initialization: Obtain the L-hop neighboring nodes of $\mathcal{G}_i$, denoted as $\{v_{k+1}, \cdots, v_r\}$. Then the set of players is $P' = \{\mathcal{G}_i, v_{k+1}, \cdots, v_r\}$.

for $i = 1$ **to** T **do**
  Sampling a coalition set S_i from $P' \setminus \{\mathcal{G}_i\}$.
  Set nodes from $V \setminus (S_i \cup \{\mathcal{G}_i\})$ with zero features and feed to the GNNs $f(\cdot)$ to obtain $f(S_i \cup \{\mathcal{G}_i\})$.
  Set nodes from $V \setminus S_i$ with zero features and feed to the GNNs $f(\cdot)$ to obtain $f(S_i)$.
  Then $m(S_i, \mathcal{G}_i) = f(S_i \cup \{\mathcal{G}_i\}) - f(S_i)$.
end for
Return: $\text{Score}(f(\cdot), \mathcal{G}, \mathcal{G}_i) = \frac{1}{T} \sum_{t=1}^{T} m(S_i, \mathcal{G}_i)$.

3.5. Graph Inspired Efficient Computations

In graph neural networks, the new features of a target node are obtained by aggregating information from a limited neighboring region. Assuming there are L layers of GNN in the graph model $f(\cdot)$, then only the neighboring nodes within L-hops are used for information aggregation. Note that the information aggregation schema can be considered as interactions between different graph structures. Hence, the subgraph $\mathcal{G}_i$ mostly interacts with the neighbors within L-hops. Based on such observations, we propose to compute the Shapley value of $\mathcal{G}_i$ by only considering its L-hop neighboring nodes. Specifically, assuming there are r ($r \leq m - k$) nodes within L-hop neighboring of subgraph $\mathcal{G}_i$, we denote these nodes as $\{v_{k+1}, \cdots, v_r\}$. Then the new set of players we need to consider is represented as $P' = \{\mathcal{G}_i, v_{k+1}, \cdots, v_r\}$. By incorporating P', the Shapley value of $\mathcal{G}_i$ can be defined as

$$\phi(\mathcal{G}_i) = \sum_{S \subseteq P' \setminus \{\mathcal{G}_i\}} \frac{|S|! \, (|P'| - |S| - 1)!}{|P'|!} m(S, G_i). \quad (8)$$

However, since graph data are complex that different nodes have variable numbers of neighbors, then P' may still contain a large number of players, thus affecting the efficiency of computation. Hence, in our SubgraphX, we further incorporate the Monte-Carlo sampling (Štrumbelj & Kononenko, 2014) to compute $\phi(\mathcal{G}_i)$. Specifically, for sampling step i, we sample a coalition set S_i from the player set $P' \setminus \{\mathcal{G}_i\}$ and compute its marginalized contribution $m(S_i, \mathcal{G}_i)$. Then

the averaged contribution score for multiple sampling steps is regarded as the approximation of $\phi(\mathcal{G}_i)$. Formally, it can be mathematically written as

$$\phi(\mathcal{G}_i) = \frac{1}{T} \sum_{t=1}^{T} (f(S_i \cup \{\mathcal{G}_i\}) - f(S_i)), \quad (9)$$

where T is the total sampling steps. In addition, to compute the marginalized contribution, we follow a zero-padding strategy. Specifically, to compute $f(S_i \cup \{\mathcal{G}_i\})$, we consider the nodes $V \setminus (S_i \cup \{\mathcal{G}_i\})$ which are not belonging to the coalition or the subgraph and set their node features to all zeros. Then we feed the new graph to the GNNs $f(\cdot)$ and use the predicted probability as $f(S_i \cup \{\mathcal{G}_i\})$. Similarly, we can compute $f(S_i)$ by setting nodes $V \setminus S_i$ with zero features and feeding to the GNNs. It is noteworthy that we only perturb the node features instead of removing the nodes from the input graph because graphs are very sensitive to structural changes (Schlichtkrull et al., 2021b). Finally, we conclude the computation steps of our proposed SubgraphX in Algorithm 1 and 2. Note that N_{min} determines the stop condition in MCTS, and we can select subgraphs with specific sizes from internal nodes of the search tree as needed.

3.6. SubgraphX for Generic Graph Tasks

We have described our proposed SubgraphX using graph classification models as an example. It is noteworthy that our SubgraphX can be easily generalized to explain graph models on other tasks, such as node classification and link prediction. For node classification models, the explanation target is the prediction of a single node v_i given the input graph $\mathcal{G}$. Assuming there are L layers in the GNN models, the prediction of v_i only relies on its L-hop computation graph, denoted as $\mathcal{G}_c$. Then instead of searching from the

Table 1. Statistics and properties of five datasets.

	Dataset				
	MUTAG	BBBP	GRAPH-SST2	BA-2MOTIFS	BA-SHAPE
# of Edges (avg)	19.79	25.95	9.20	25.48	2055
# of Nodes (avg)	17.93	24.06	10.19	25.0	700
# of Graphs	188	2039	70042	1000	1
# of Classes	2	2	2	2	4

input graph $\mathcal{G}$, our SubgraphX sets $\mathcal{G}_c$ as the corresponding graph of the search tree root $\mathcal{N}_0$. In addition, when computing the marginalized contributions, the zero-padding strategy should exclude the target node v_i. Meanwhile, for link prediction tasks, the explanation target is the prediction of a single link (v_i, v_j). Then the root of the search tree corresponds to the L-hop computation graph of node v_i and v_j. Similarly, the zero-padding strategy ignores the v_i and v_j when perturbing node features. Note that our SubgraphX treats the GNNs as black boxes during the explanation stage and only needs to access the inputs and outputs. Hence, our proposed SubgraphX can be applied to a general family of GNN models, including but not limited to GCNs (Kipf & Welling, 2017), GATs (Veličković et al., 2018), GINs (Xu et al., 2019), and Line-Graph NNs (Chen et al., 2018b).

4. Experimental Studies

4.1. Datasets and Experimental Settings

We conduct extensive experiments on different datasets and GNN models to demonstrate the effectiveness of our proposed method. The statistics and properties of the datasets are reported in Table 1. We evaluate our SubgraphX with five datasets for both graph classification and node classification tasks, including synthetic data, biological data, and text data. We summarize these datasets as below:

- MUTAG (Debnath et al., 1991) and BBBP (Wu et al., 2018) are molecular datasets for graph classification tasks. In these datasets, each graph represents a molecule while nodes are atoms and edges are bonds. The labels are determined by the chemical functionalities of molecules.
- Graph-SST2 (Yuan et al., 2020c) is sentiment graph dataset for graph classification. It converts text sentences to graphs with Biaffine parser (Gardner et al., 2018) that nodes denote words and edges represent the relationships between words. Note that node embeddings are initialized as the pre-trained BERT word embeddings (Devlin et al., 2019). Each graph is labeled by its sentiment, which can be positive or negative.
- BA-2Motifs is a synthetic graph classification dataset. Each graph contains a based graph generated by *Barabási-Albert* (BA) model, which is connected with a house-like

motif or a five-node cycle motif. The graphs are labeled based on the type of motifs. All node embeddings are initialized as vectors containing all 1s.
- BA-Shape is a synthetic node classification dataset. Each graph contains a base BA graph and several house-like five-node motifs. The node labels are determined by the memberships and locations of different nodes. All node embeddings are initialized as vectors containing all 1s.

We explore three variants of GNNs on these datasets, including GCNs, GATs, and GINs. All GNN models used in our experimental studies are trained to obtain reasonable performance. Then we compare our SubgraphX with several baselines, including MCTS_GNN, GNNExplainer (Ying et al., 2019), PGExplainer (Luo et al., 2020). Note that GNNExplainer and PGExplainer represent the state-of-the-art methods for GNN explanations. Here MCTS_GNN denotes the method using MCTS to explore subgraphs but directly employing the GNN predictions of these subgraphs as the scoring function. We wish to mention that all methods are compared with a fair setting. We use the same number to control the maximum number of nodes in the explanations for all methods.

We conduct our experiments using one Nvidia V100 GPU on an Intel Xeon Gold 6248 CPU. Our implementations are based on Python 3.7.6, PyTorch 1.6.0, and Torch-geometric 1.6.3. For our proposed SubgraphX and other algorithms with MCTS, the MCTS iteration number M is set to 20. To explore a suitable trade-off between exploration and exploitation, we set the hyperparameter λ in Eq.(3) to 5 for Graph-SST2 (GATs) and BBBP (GCNs) models, and 10 for other models. Since all GNN models contain 3 network layers, we consider 3-hop computational graphs to compute Shapley values for our SubgraphX. For the Monte-Carlo sampling in our SubgraphX, we set the Monte-Carlo sampling steps T to 100 for all datasets. For MCTS[†], we set Monte-Carlo sampling steps to 1000 to obtain good approximations since it samples from all nodes in a graph. More details regarding the datasets, trained GNN models, and experimental settings can be found in Supplementary Section A. Our code and data are now publicly available in the DIG library (Liu et al., 2021)[1].

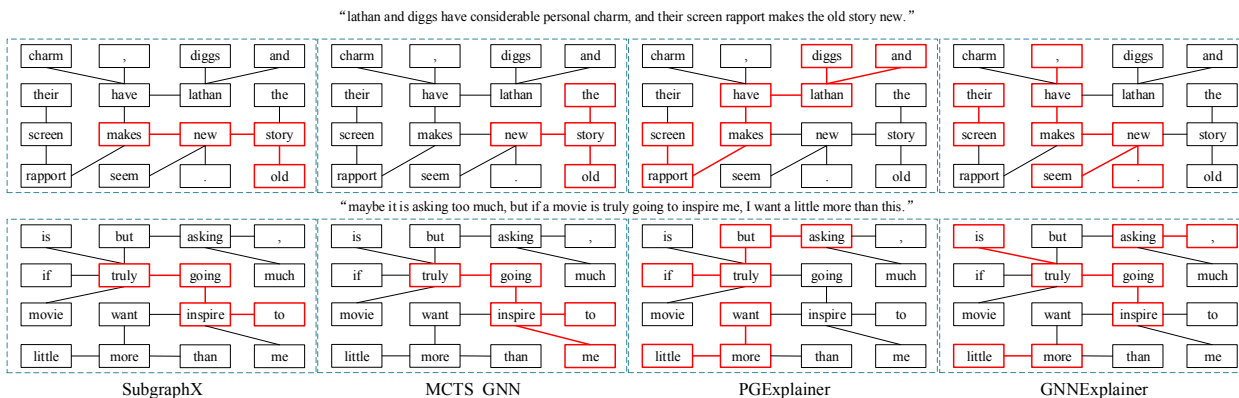
[1]https://github.com/divelab/DIG

Figure 2. Explanation results on the Graph-SST2 dataset with a GAT graph classifier. The input sentences are shown on the top of explanations. Note that some "unimportant" words are ignored for simplicity. The first row shows explanations for a correct prediction and the second row reports the results for an incorrect prediction.
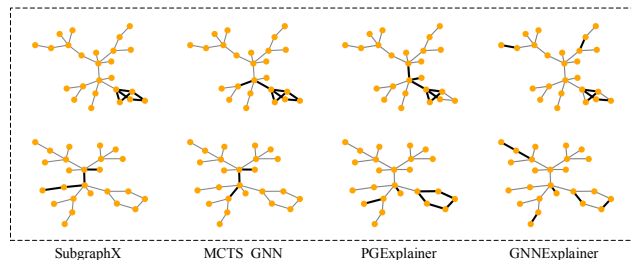

Figure 3. Explanation results on the BA-2Motifs dataset with a GCN graph classifier. The first row shows explanations for a correct prediction and the second row reports the results for an incorrect prediction.

Figure 4. Explanation results on the MUTAG dataset with a GIN graph classifier. We show the explanations for two correct predictions. Here Carbon, Oxygen, and Nitrogen are shown in yellow, red, and blue, respectively.

4.2. Explanations for Graph Classification Models

We first visually compare our SubgraphX with the other baselines using graph classification models. The results are reported in Figure 3, 4, and 2 where important substructures are shown in the bold.

The explanation results of the BA-2Motifs dataset are visualized in Figure 3. We use the GCNs as the graph classifier and report explanations for both correct and incorrect predictions. Since it is a synthetic dataset, we may consider the motifs as reasonable approximations of explanation ground truth. In the first row, the model prediction is correct and our SubgraphX can precisely identify the house-like motif as the most important subgraph. In the second row, our SubgraphX explains the incorrect prediction that the GNN model cannot capture the five-node cycle motif as the important structure, and hence the prediction is wrong. For both cases, our SubgraphX can provide better visual explanations since our method can precisely identifies the succinct subgraphs that can reasonably explain the predictions. In addition, our explanations are connected subgraphs while
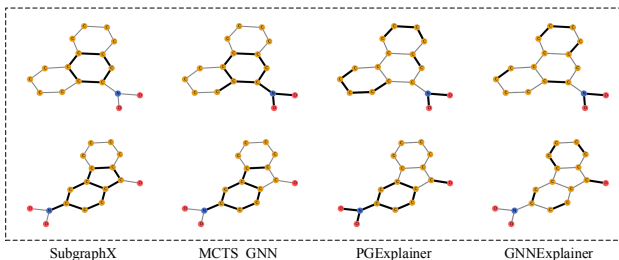
PGExplainer and GNNExplainer identify discrete edges.

We also show the explanation results of the MUTAG dataset in Figure 4. Note that GINs are employed as the graph classification model to be explained. Since the MUTAG dataset is a real-world dataset and there is no ground truth for explanations, we evaluate the explanation results based on chemical domain knowledge. The graphs in MUTAG are labeled based on the mutagenic effects on a bacterium. It is known that carbon rings and NO_2 groups tend to be mutagenic (Debnath et al., 1991). We study whether the explanations provided by different methods can match the carbon rings and NO_2 groups identified in chemistry. In both examples, the predictions are "mutagenic" and our SubgraphX successfully and precisely identifies the carbon rings as important subgraphs. Meanwhile, the MCTS_GNN can capture the key subgraphs but include several additional edges. The results of the PGExplainer and GNNExplainer still contain several discrete edges.

For the dataset Graph-SST2, we employ GATs as the graph model and report the results in Figure 2. In the first row,
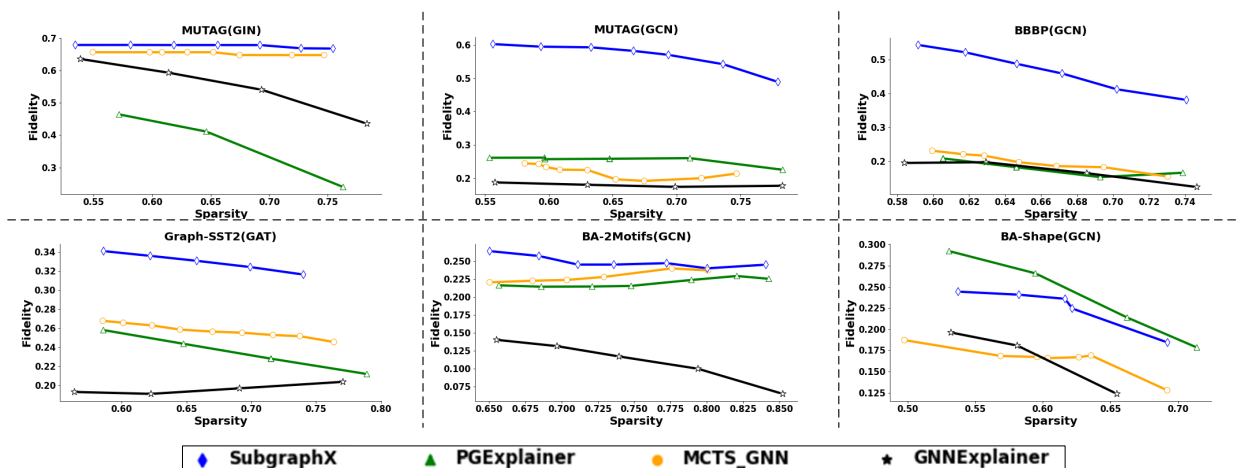
Figure 5. The quantitative studies for different explanation methods. Note that since the Sparsity scores cannot be fully controlled, we compare different methods with Fidelity scores under similar similar levels of Sparsity.

the prediction is correct and the label is positive. Both our SubgraphX and the MCTS_GNN can find word phrases with positive semantic meaning, such as "makes old story new", which can reasonably explain the prediction. The explanations provided by PGExplainer and GNNExplainer are, however, less semantically related. In the second row, the input is negative but the prediction is positive. All methods except PGExplainer can explain the decision that the GNN model regards positive phrases "truly going to inspire" as important, thus yielding a positive but incorrect prediction. It is noteworthy that our method tends to include fewer neural words, such as "the", "me", and "screen", etc.

Overall, our SubgraphX can explain both correct and incorrect predictions for different graph data and GNN models. Our explanations are more human-intelligible than comparing methods. More results for graph classification models are reported in Supplementary Section B.

4.3. Explanations for Node Classification Models

We also compare different methods on the node classification tasks. We use the BA-Shape dataset and train a GCN model to perform node classification. The visualization results are reported in Figure 6 where the important substructures are shown in bold. We can verify if the explanations are consistent with the rules (the motifs) to label different nodes. For both examples, the target nodes are correctly classified. Obviously, our SubgraphX is precisely targeting the motifs as the explanations, which is reasonable and promising. For other methods, their explanations only cover partial motifs and include other structures. More results are reported in Supplementary Section C.
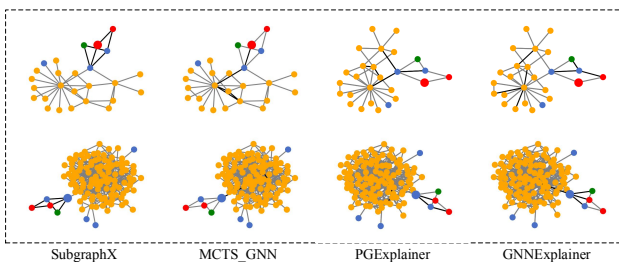

Figure 6. Explanation results on the BA-Shape dataset. The target node is shown in a larger size. Different colors denote node labels.

4.4. Quantitative Studies

While visualizations are important to evaluate different explanation methods, human evaluations may not be accurate due to the lack of ground truths. Hence, we further conduct quantitative studies to compare these methods. Specifically, we employ the metrics Fidelity and Sparsity to evaluate explanation results (Pope et al., 2019; Yuan et al., 2020c). The Fidelity metric measures whether the explanations are faithfully important to the model's predictions. It removes the important structures from the input graphs and computes the difference between predictions. In addition, the Sparsity metric measures the fraction of structures that are identified as important by explanation methods. Note that high Sparsity scores mean smaller structures are identified as important, which can affect the Fidelity scores since smaller structures (high Sparsity) tend to be less important (low Fidelity). Hence, for fair comparisons, we compare different methods using Fidelity under similar levels of Sparsity. The results are reported in Figure 5 where we plot the curves of Fidelity scores with respect to the Sparsity scores. Obviously, for five out of six experiments, our proposed

Table 2. Efficiency studies of different methods.

Method	MCTS*	MCTS†	SubgraphX	GNNExplainer	PGExplainer
TIME	>10 hours	865.4 ± 1.6s	77.8 ± 3.8s	16.2 ± 0.2s	0.02s (Training 362s)
FIDELITY	N/A	0.53	0.55	0.19	0.18

method outperforms the comparing methods significantly and consistently under different sparsity levels. For the BA-Shape (GCN) experiment, our SubgraphX obtains slightly lower but still competitive Fidelity scores compared with the PGExplainer. Overall, such results indicate that the explanations of our method are more faithful and important to the GNN models. More details of evaluation metrics are introduced in Supplementary Section A.

4.5. Efficiency Studies

Finally, we study the efficiency of our proposed method. For 50 graphs with an average of 24.96 nodes from the BBBP dataset, we show the averaging time cost to obtain explanations for each graph. We repeat the experiments 3 times and report the results Table 2. Here MCTS* denotes the baseline that follows Eq. (8) to compute Shapley values. Compared with our SubgraphX, the difference is the usage of Monte Carlo sampling. In addition, MCTS† indicates the baseline computing Shapley values with Monte Carlo sampling but without our proposed approximation schemes. Specifically, MCTS† samples coalition sets from the player set P instead of the reduced set P'. First, the time cost of MCTS* is extremely high since it needs to enumerate all possible coalition sets. Next, compared with MCTS†, our SubgraphX is 11 times faster while the obtained explanations have similar Fidelity scores. It demonstrates our approximation schemes are both effective and efficient. Even though our method is slower than GNNExplainer and PGExplainer, the Fidelity scores of our explanations are 300% higher than theirs. Furthermore, the PGExplainer requires to train its model using the whole dataset. While offline model training cost is not directly comparable, the additional and significant time cost may be an issue when the dataset is large-scale. Considering our explanations are with higher-quality and more human-intelligible, we believe such time complexity is reasonable and acceptable.

4.6. Study of Pruning Actions

Finally, we discuss the pruning actions in our MCTS. For the graph associated with each non-leaf tree search node, we perform node pruning to obtain its children subgraphs. Specifically, when a node is removed, all edges connected with it are also removed. In addition, if multiple disconnected subgraphs are obtained after removing a node, only the largest subgraph is kept. Instead of exploring all possible

Table 3. The studies of different pruning strategies.

Method	Time	Fidelity
LOW2HIGH	107.24s	0.66149
HIGH2LOW	21.52s	0.61046

node pruning actions, we explore two strategies: Low2high and High2low. First, Low2high arranges the nodes based on their node degrees from low to high and only considers the pruning actions corresponding to the first k low degree nodes. Meanwhile, High2low arranges the nodes in order from high degree to low degree and only considers the first k high degree nodes for pruning. Intuitively, High2low is more efficient but may ignore the optimal solutions. In this work, we employ the High2low strategy for BA-Shape(GCNs), and Low2high strategy for other models, and set the k to 12 for all the datasets. We conduct experiments to analyze these two pruning strategies for our SubgraphX algorithm and show the average time cost and Fidelity score in Table 3. Specifically, we randomly select 50 graphs from the BBBP datasets with an average node number of 24.96, which is the same in Section 4.5. In addition, we set Monte-Carlo sampling steps T to 100, and select the subgraphs with the highest Shapley values and contain less than 15 nodes to calculate the Fidelity. Obviously, High2low is 5 times faster than Low2high but the Fidelity scores are inferior.

5. Conclusions

While considerable efforts have been devoted to study the explainability of GNNs, none of existing methods can explain GNN predictions with subgraphs. We argue that subgraphs are building blocks of complex graphs and are more human-intelligible. To this end, we propose the SubgraphX to explain GNNs by identifying important subgraphs explicitly. We employ the Monte Carlo tree search algorithm to efficiently explore different subgraph. For each subgraph, we propose to employ Shapley values to measure its importance by considering the interactions among different graph structures. To expedite computations, we propose efficient approximation schemes to compute Shapley values by considering interactions only within the information aggregation range. Experimental results show our SubgraphX obtain higher-quality and more human-intelligible explanations while keeping time complexity acceptable.

References

Alon, U. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461, 2007.

Alon, U. *An introduction to systems biology: design principles of biological circuits*. CRC press, 2019.

Baldassarre, F. and Azizpour, H. Explainability techniques for graph convolutional networks. In *International Conference on Machine Learning (ICML) Workshops, 2019 Workshop on Learning and Reasoning with Graph-Structured Representations*, 2019.

Chen, J., Song, L., Wainwright, M. J., and Jordan, M. I. L-shapley and c-shapley: Efficient model interpretation for structured data. In *International Conference on Learning Representations*, 2018a.

Chen, Z., Li, L., and Bruna, J. Supervised community detection with line graph neural networks. In *International Conference on Learning Representations*, 2018b.

Debnath, A. K., Lopez de Compadre, R. L., Debnath, G., Shusterman, A. J., and Hansch, C. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry*, 34 (2):786–797, 1991.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Du, M., Liu, N., Song, Q., and Hu, X. Towards explanation of dnn-based prediction with guided feature inversion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1358–1367, 2018.

Gao, H. and Ji, S. Graph U-Nets. In *international conference on machine learning*, pp. 2083–2092. PMLR, 2019.

Gao, H., Wang, Z., and Ji, S. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1416–1424, 2018.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N. F., Peters, M., Schmitz, M., and Zettlemoyer, L. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pp. 1–6, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-2501. URL https://www.aclweb.org/anthology/W18-2501.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pp. 1263–1272. PMLR, 2017.

Huang, Q., Yamada, M., Tian, Y., Singh, D., Yin, D., and Chang, Y. Graphlime: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216*, 2020.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2016.

Jin, W., Barzilay, R., and Jaakkola, T. Multi-objective molecule generation using interpretable substructures. In *International Conference on Machine Learning*, pp. 4849–4859. PMLR, 2020.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Kuhn, H. W. and Tucker, A. W. *Contributions to the Theory of Games*, volume 2. Princeton University Press, 1953.

Liu, M., Luo, Y., Wang, L., Xie, Y., Yuan, H., Gui, S., Yu, H., Xu, Z., Zhang, J., Liu, Y., Yan, K., Liu, H., Fu, C., Oztekin, B., Zhang, X., and Ji, S. DIG: A turnkey library for diving into graph deep learning research. *arXiv preprint arXiv:2103.12608*, 2021.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pp. 4765–4774, 2017.

Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., and Zhang, X. Parameterized explainer for graph neural network. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19620–19631. Curran Associates, Inc., 2020.

Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., and Alon, U. Network motifs: simple building blocks of complex networks. *Science*, 298(5594): 824–827, 2002.

Pope, P. E., Kolouri, S., Rostami, M., Martin, C. E., and Hoffmann, H. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10772–10781, 2019.

Schlichtkrull, M. S., Cao, N. D., and Titov, I. Interpreting graph neural networks for {nlp} with differentiable edge masking. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=WznmQa42ZAx.

Schlichtkrull, M. S., Cao, N. D., and Titov, I. Interpreting graph neural networks for {nlp} with differentiable edge masking. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=WznmQa42ZAx.

Schnake, T., Eberle, O., Lederer, J., Nakajima, S., Schütt, K. T., Müller, K.-R., and Montavon, G. Xai for graphs: Explaining graph neural network predictions by identifying relevant walks. *arXiv preprint arXiv:2006.03589*, 2020.

Schwarzenberg, R., Hübner, M., Harbecke, D., Alt, C., and Hennig, L. Layerwise relevance visualization in convolutional text graph classifiers. *arXiv preprint arXiv:1909.10911*, 2019.

Shen-Orr, S. S., Milo, R., Mangan, S., and Alon, U. Network motifs in the transcriptional regulation network of escherichia coli. *Nature genetics*, 31(1):64–68, 2002.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net. *International Conference on Learning Representations*, 2015.

Štrumbelj, E. and Kononenko, I. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Vu, M. N. and Thai, M. T. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. In *Advances in neural information processing systems*, 2020.

Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., and Yu, P. S. Heterogeneous graph attention network. In *The World Wide Web Conference*, pp. 2022–2032, 2019.

Wang, Z., Liu, M., Luo, Y., Xu, Z., Xie, Y., Wang, L., Cai, L., and Ji, S. Advanced graph and sequence neural networks for molecular property prediction and drug discovery. *arXiv preprint arXiv:2012.01981*, 2020.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Yang, F., Pentyala, S. K., Mohseni, S., Du, M., Yuan, H., Linder, R., Ragan, E. D., Ji, S., and Hu, X. Xfake: explainable fake news detector with visualizations. In *The World Wide Web Conference*, pp. 3600–3604, 2019.

Ying, Z., Bourgeois, D., You, J., Zitnik, M., and Leskovec, J. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in neural information processing systems*, pp. 9244–9255, 2019.

Yuan, H. and Ji, S. Structpool: Structured graph pooling via conditional random fields. In *International Conference on Learning Representations*, 2020.

Yuan, H. and Ji, S. Node2seq: Towards trainable convolutions in graph neural networks. *arXiv preprint arXiv:2101.01849*, 2021.

Yuan, H., Chen, Y., Hu, X., and Ji, S. Interpreting deep models for text analysis via optimization and regularization methods. In *AAAI-19: Thirty-Third AAAI Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence, 2019.

Yuan, H., Cai, L., Hu, X., Wang, J., and Ji, S. Interpreting image classifiers by generating discrete masks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020a.

Yuan, H., Tang, J., Hu, X., and Ji, S. XGNN: Towards model-level explanations of graph neural networks. KDD '20, pp. 430–438, New York, NY, USA, 2020b. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403085. URL https://doi.org/10.1145/3394486.3403085.

Yuan, H., Yu, H., Gui, S., and Ji, S. Explainability in graph neural networks: A taxonomic survey. *arXiv preprint arXiv:2012.15445*, 2020c.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Zhang, M., Cui, Z., Neumann, M., and Chen, Y. An end-to-end deep learning architecture for graph classification. In *AAAI*, volume 18, pp. 4438–4445, 2018.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2921–2929, 2016.