GraphLIME: Local Interpretable Model Explanations for Graph Neural Networks

Qiang Huang[1], Makoto Yamada[2,3], Yuan Tian[1], Dinesh Singh[3], Dawei Yin[4], Yi Chang[1]

[1]Jilin University, [2]Kyoto University, [3]RIKEN AIP, [4]JD.com

huangqiang18@mails.jlu.edu.cn, myamada@i.kyoto-u.ac.jp, yuantian@jlu.edu.cn
dinesh.singh@riken.jp, yindawei@acm.org, yichang@jlu.edu.cn

January 20, 2020

Abstract

Graph structured data has wide applicability in various domains such as physics, chemistry, biology, computer vision, and social networks, to name a few. Recently, graph neural networks (GNN) were shown to be successful in effectively representing graph structured data because of their good performance and generalization ability. GNN is a deep learning based method that learns a node representation by combining specific nodes and the structural/topological information of a graph. However, like other deep models, explaining the effectiveness of GNN models is a challenging task because of the complex nonlinear transformations made over the iterations. In this paper, we propose GraphLIME, a local interpretable model explanation for graphs using the Hilbert-Schmidt Independence Criterion (HSIC) Lasso, which is a nonlinear feature selection method. GraphLIME is a generic GNN-model explanation framework that learns a nonlinear interpretable model locally in the subgraph of the node being explained. More specifically, to explain a node, we generate a nonlinear interpretable model from its N-hop neighborhood and then compute the K most representative features as the explanations of its prediction using HSIC Lasso. Through experiments on two real-world datasets, the explanations of GraphLIME are found to be of extraordinary degree and more descriptive in comparison to the existing explanation methods.

1 Introduction

Deep Neural Network (DNN) is essentially a new machine learning algorithm based on discriminant model. DNN can model for complex and nonlinear problems and learn the underlying features of data to obtain more abstract features, which can improve the model's capability for prediction or classification. It has a large number of applications such as image recognition, automatic speech recognition, disease diagnosis, etc.

Although the high degree of nonlinearity gives DNNs a powerful model representation capability, DNN is a black-box model, which make the model and its predictions hard to be interpreted. Also, people usually will not use a model or a prediction if they do not trust it. For example, if there is a medical diagnosis system that has high accuracy but can not present faithful explanations for its decisions, doctors will not use it. Therefore, it is very important to explore what a DNN model learns from data and why it takes a particular decision in a way that humans can understand. Recent works aimed at interpreting general DNNs mainly focus on two research routes. One approach is to locally approximate models with a simple and interpretable model such as linear regression, which can itself be probed for explanations [1, 2]. The other one is to examine models for relevant components such as identifying the most representative features in the input data [3, 4, 5] or influential input instances [6, 7].

In many real-world cases, there are considerable amounts of data without regular spatial structures, called non-Euclidean data, and they can be naturally represented as a graph, for example, graph data extracted from social networks, citation
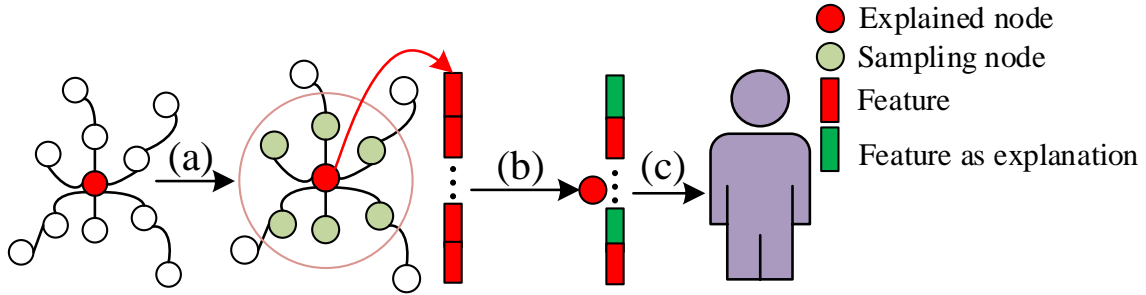
Figure 1: (a) Sample neighbors of the explained red node; (b) Obtain finite numbers of features as explanations (the green part of the features of the red node); (c) Present explanations to users.

networks, electronic transactions, protein structures, molecular structures, and so on. Thus, modeling and analyzing graph data is a challenging task since it needs to combine both feature information of nodes as well as graph information together.

Currently, Graph Neural Networks (GNNs) are widely used because of their powerful modeling capability with regard to graphs. GNNs use neural networks to incorporate the feature information of nodes in a graph and as well as the structure information and pass these messages through the edges of the graph in non-Euclidean domains. However, GNNs are notoriously difficult to interpret, and their predictions are hard to explain, similar to that in DNNs. It is important to develop an interpretation method for GNNs because it can improve the transparency of a GNN model and contribute to getting humans to trust the model. Although there exists a large number of interpretation methods designed for DNNs, these methods are not suited for GNN-model interpretation because they do not explicitly use the graph information but perform in Euclidean domains.

Recently, GNNexplainer [8] was proposed. It can find the subgraph and select features of the explained node as explanations, but it mainly focuses on graph structures and not on finding useful features. An alternative idea is to use LIME [2], which uses a linear explanation model to find features as explanations for GNN. However, the performance of LIME can be poor, because LIME does not take the graph structure information into account. Moreover, if the underlying model is highly nonlinear even in the locality of the prediction, such as that in the case of GNN models, a linear explanation model may not be able to produce faithful explanations.

In this paper, we propose another GNN explanation method based on LIME [2] in a nonlinear manner. More specifically, we propose GraphLIME, a model-agnostic and nonlinear approach for providing locally faithful explanations for GNN-based models in a subgraph, whose procedure is shown briefly in Figure 1. It samples N-hop network neighbors of the node being explained and locally captures the nonlinear dependency between features and predictions based on HSIC Lasso. GraphLIME is a simple yet effective interpretation method, which locally finds the most representative features as explanations in a nonlinear manner. Note that GraphLIME can be treated as a nonlinear graph-variant of the LIME method [2], which considers perturbation near the node being explained and applies a linear interpretable model. Through experiments on two real-world datasets, it is shown that the explanations of GraphLIME are of extraordinary degree and more descriptive in comparison to the existing explanation methods.

2 Related Work

In this section, we briefly review related work from two aspects. The first one is the recent development of GNNs, the second one is the review of interpretation methods for neural models.

2.1 Graph Neural Networks

Recently, there have been many studies on extending deep learning approaches for graph data. GNNs [9] recursively incorporate the node feature information and structure information of a graph from the neighborhood into a neural network and obtain node embeddings for subsequent machine learning tasks. GraphSAGE [10] is a spatial based and general inductive framework that can generate a node's embeddings by essentially assembling a node's neighborhood information from the node's local neighborhood, and its aggregation function must be invariant to permutations of node orderings. Novel control variate-based stochastic approximation algorithms for GCN [11] have been proposed, which utilize the historical activations of nodes as a control variate and allow sampling an arbitrarily small neighbor size. Graph Attention Network (GAT) [12] is one type of GNNs, which seeks an aggregation function to fuse the neighboring nodes, random walks, and candidate models in graphs to learn a new representation. The main idea of GATs is to employ attention mechanisms that assign larger weights to the more important nodes, walks, or models. Gated Attention Network (GaAN) [13] is a new network architecture that uses a convolutional sub-network to compute a soft gate at each attention head to control its importance, and the gated attention can modulate the amount of attended content via the introduced gates. Graph Attention Model (GAM) [14] processes only a portion of the graph by adaptively selecting a sequence of "informative" nodes and is equipped with an external memory component, which allows it to integrate the information gathered from different parts of the graph. Dual Graph Convolutional Network [15] was proposed as a simple and scalable semi-supervised learning method for graph-structured data in which only a very small portion of the training data are labeled, and it jointly considers local and global consistency assumptions.

2.2 Interpretability and Explanations for Neural Networks

There are two main families of models to provide interpretability and explanations for neural models. The first family of models focuses on finding a simple and proxy model to interpret the model being explained in a model-agnostic way. For example, LIME [2] has been proposed to explain the predictions of any classifier by learning an interpretable linear model locally around the prediction. Black Box Explanations through Transparent Approximations (BETA) [1] is another model-agnostic framework [1] to explain the behavior of the model being expalined by simultaneously optimizing for fidelity to the original model and interpretability of the explanation. DeepRED [16] was proposed to extract rules from deep neural networks and generate explanations for predictions, and ANN-DT [17] was proposed to extract binary decision trees from neural networks, it is an interpretable model to interpret the main model.

The second family of models focuses on the relevant aspects of computation in the neural model being explained. Erhan et al. [18] proposed to inspect feature gradients to find good qualitative interpretations of high-level features represented by neural models. DeepLIFT (Deep Learning Important FeaTures) [19] was proposed to enable interpretability of neural networks by comparing the activation of each neuron to its 'reference activation' and assigning contribution scores according to the difference. The method proposed in [5] uses two fundamental axioms and a standard gradient operator to attribute the prediction of a deep network to its input features. It is worth noting that these methods produce a "saliency map" [20], which reveals the important features of input data. However, recent works have shown that saliency map will give rise to misleading results in some instances [21] and can lead to issues such as gradient saturation in some cases. This issue is especially grievous for discrete inputs such as graph adjacency matrices. Therefore, these methods are not suitable for interpreting GNNs, and this is what we do in this paper. Instead of creating surrogate models, which are inherently interpretable, some methods has been proposed to identify the patterns of input data and find influential samples for relevant information [6, 7]. However, few studies have focused on the structural information of graphs with these kinds of interpretation methods, although the relations between nodes in the graph are critical to the predictions from deep models. Recently, Ying et al. [8] proposed to utilize mutual information to find a subgraph with associated features for interpreting GNN models; however, this method depends on discrete optimization, and the optimization is hard and only consider the predicted label of the node being explained. The proposed GraphLIME method utilizes predicted labels from both the node being explained and its neighbors, which enables it to capture more local information around the node and present a finite number of features as explanations in an intuitional way.

3 GraphLIME

In this paper, we propose Local Interpretable Model Explanation for Graph Neural Networks (GraphLIME), which is a nonlinear model-agnostic explanation framework for GNNs. The goal of the proposed explanation framework is to identify a nonlinear interpretable model over the neighbors of the node being explained in a subgraph, which are locally faithful to the node's prediction, and interpret the GNN model, which is powerful but hard to explain in an intuitional manner.

3.1 Formulation of GraphLIME Explainer

Formally, let $\mathcal{X} \subset \mathbb{R}^d$ be the domain of a vector $\boldsymbol{x}$. For each node v_i in a graph, we have an associated feature vector $\boldsymbol{x}_i$, where each feature is extracted from the real world and can be understood by humans. We define an explanation as a model $g \in G$, where G is a class of *interpretable* models such as linear regressions, decision trees etc. In other words, a model $g \in G$ can provide intuitional interpretations in an interpretable manner. The explanation model class G used in this paper is Hilbert-Schmidt Independence Criterion Lasso (HSIC Lasso) [22, 23], which is a kernel based nonlinear interpretable feature selection algorithm. We will discuss more details about this in a later subsection. Let the domain of the explanation model $g \in G$ also be $\mathcal{X} \subset \mathbb{R}^d$, which means that g acts over the original features that are understandable.

Let the GNN model being explained be denoted by $f : \mathbb{R}^d \to \mathbb{R}$. In a classification problem, $f(\boldsymbol{x}_i)$ denotes the probability (or binary indicator) that the instance $\boldsymbol{x}_i$ belongs to a certain class[1]. Let v denote the node whose prediction needs to be explained, $\boldsymbol{X}_n \in \mathbb{R}^{n \times d}$ represents the sampling information matrix that can capture the locality of the explained node v. Then, we obtain an interpretable model $g \in G$ using the local information matrix $\boldsymbol{X}_n$ of v to approximate f, and generate the locally faithful explanations based on g.

Given a GNN model f, the node v being explained, the sampling local information matrix $\boldsymbol{X}_n$ of v, and an interpretable explanation model g, the explanations for the explained v are obtained as follows:

$$\zeta(v) = \underset{g \in G}{\mathrm{argmin}} \; g(f, \boldsymbol{X}_n), \tag{1}$$

where $\zeta(v)$ is the set of features as explanations of the node being explained v.

Actually, different explanation models and sampling methods can be used in this formulation. Here, we focus on the nonlinear model for explanations, which can capture the nonlinear dependency between features and outputs and find the most representative features, and on the N-hop network neighbors sampling method to obtain the locally faithful explanations in a subgraph. Next, we will discuss the details about the sampling method and nonlinear explanation model of GraphLIME.

3.2 Sampling for Local Exploration by N-hop Network Neighbors

In a graph, a node's local information is decided by its neighbors in a subgraph; hence, the explainers for a GNN model in a graph should not only focus on the features of the given node but also the correlations between the features of its neighbouring nodes, because a given target node needs to aggregate information from its neighboring nodes and then make a prediction. In order to learn the local behavior of a GNN model being explained, we consider N-hop network neighbors to sample the neighboring nodes of a given node. A node v's N-hop network is defined as the network formed by v and the nodes whose distance from v is within N hops and links. By performing N-hop network sampling, we obtain:

$$\boldsymbol{X}_n = \{\boldsymbol{x}_i | v_i \in \mathcal{S}_N\},$$

where $\mathcal{S}_N$ is the set of N-hop neighbors of the explaining node v, and $\boldsymbol{x}_i$ is the corresponding associated feature vector of v_i. Moreover, we can also obtain prediction $y_i = f(\boldsymbol{x}_i)$ for a given GNN model f, which can be used as the label in the explanation model g. We now present a nonlinear algorithm as the explanation model in the proposed framework GraphLIME.

---

[1]For multiple classes, we explain each class separately.

3.3 Nonlinear Explanation Model: HSIC LASSO

We consider a feature-wise kernelized nonlinear method called Hilbert-Schmidt Independence Criterion Lasso (HSIC Lasso) [22, 23] as the explanation model for GNN models. The HSIC Lasso is a "supervised" nonlinear feature selection method. Given a GNN model f, a node v whose prediction needs to be explained, the neighbor set $\boldsymbol{X}_n$ of node v, and "supervised" paired N-hop neighboring nodes $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{x}_i \in \boldsymbol{X}_n$, the probability $y_i = f(\boldsymbol{x}_i)$ that x_i belongs to a certain class is used as the *label* of the HSIC Lasso explanation model. For each prediction being explained, the HSIC Lasso optimization problem for an explanation model $g \in G$ is given as

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \quad \frac{1}{2} \|\bar{\boldsymbol{L}} - \sum_{k=1}^{d} \beta_k \bar{\boldsymbol{K}}^{(k)}\|_F^2 + \rho\|\boldsymbol{\beta}\|_1, \quad \text{s.t.} \quad \beta_1, \dots, \beta_d \geq 0, \tag{2}$$

where $\|\cdot\|_F$ is the Frobenius norm, $\rho \geq 0$ is the regularization parameter, $\|\cdot\|_1$ is the ℓ_1 norm to enforce sparsity, $\bar{\boldsymbol{L}} = \boldsymbol{HLH}/\|\boldsymbol{HLH}\|_F$ is the normalized centered Gram matrix, $\boldsymbol{L}_{ij} = L(y_i, y_j)$ is the kernel for the output, $\boldsymbol{H} = \boldsymbol{I}_n - \frac{1}{n}\boldsymbol{1}_n\boldsymbol{1}_n^\top$ is the centering matrix, $\boldsymbol{I}_n$ is the n-dimension identity matrix, $\boldsymbol{1}_n$ is the n-dimension vector whose elements are all 1, $\bar{\boldsymbol{K}}^{(k)} = \boldsymbol{HK}^{(k)}\boldsymbol{H}/\|\boldsymbol{HK}^{(k)}\boldsymbol{H}\|_F$ is the normalized centered Gram matrix for the k-th feature, and $[\boldsymbol{K}^{(k)}]_{ij} = K(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)})$ is the kernel for the k-th dimensional input.

In this paper, we use the Gaussian kernel for both input and the predictions of $\boldsymbol{X}_n$ from the given GNN model:

$$K(\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}) = \exp\left(-\frac{(\boldsymbol{x}_i^{(k)} - \boldsymbol{x}_j^{(k)})^2}{2\sigma_x^2}\right),$$

$$L(y_i, y_j) = \exp\left(-\frac{\|y_i - y_j\|_2^2}{2\sigma_y^2}\right),$$

where σ_x and σ_y are the Gaussian kernel widths.

We employ the nonnegative least angle regression [24] to optimize the Eq. (2), and then we can obtain the coefficient vector $\boldsymbol{\beta}$ and select the top-K features as the explanations for the prediction of the node being explained based on it.

3.4 Interpretation of HSIC Lasso

Here, we present the interpretation of the nonlinear explanation model HSIC Lasso. The HSIC Lasso contains the main concepts of minimum redundancy maximum relevancy (mRMR) [25], which is a widely used classical supervised feature selection algorithm that can find non-redundant features with strong dependence on the output values. We can rewrite the first term of Eq. (2) as

$$\frac{1}{2} \|\bar{\boldsymbol{L}} - \sum_{k=1}^{d} \beta_k \bar{\boldsymbol{K}}^{(k)}\|_F^2 = \frac{1}{2} \sum_{k,m=1}^{d} \beta_k \beta_m \text{NHSIC}(\boldsymbol{f}_k, \boldsymbol{f}_m) - \sum_{k=1}^{d} \beta_k \text{NHSIC}(\boldsymbol{f}_k, \boldsymbol{y}) + \frac{1}{2} \tag{3}$$

where $\boldsymbol{f}_k$ is the feature vector corresponding to the k-th feature, $\text{NHSIC}(\boldsymbol{f}_k, \boldsymbol{y}) = \text{tr}(\bar{\boldsymbol{K}}^{(k)}\bar{\boldsymbol{L}})$ is the normalized variant of the empirical estimate of the Hibert-Schmidt independence criterion (HSIC) [26], $\text{NHSIC}(\boldsymbol{y}, \boldsymbol{y}) = 1$ is constant, and $\text{tr}(\cdot)$ is the trace operator. HSIC, which is based on a universal reproducing kernel such as the Gaussian kernel, is a non-negative function that estimates the independence between two random variables. A larger HSIC value indicates more dependency between the two variables, and it is zero if and only if the two random variables are statistically independent.

In Eq. (3), we ignore the value of $\text{NHSIC}(\boldsymbol{y}, \boldsymbol{y})$ because it is constant. We consider the values of $\text{NHSIC}(\boldsymbol{f}_k, \boldsymbol{y})$ and $\text{NHSIC}(\boldsymbol{f}_k, \boldsymbol{f}_m)$. If there is strong dependency between the k-th feature vector $\boldsymbol{f}_k$ and the output vector $\boldsymbol{y}$, the value of $\text{NHSIC}(\boldsymbol{f}_k, \boldsymbol{y})$ should be large and the corresponding coefficient β_k should also take a large value in order to minimize Eq. (2). Meanwhile, if $\boldsymbol{f}_k$ is independent of $\boldsymbol{y}$, the value of $\text{NHSIC}(\boldsymbol{f}_k, \boldsymbol{y})$ should be small so that β_k tends to be eliminated by l_1-regularizer. This property can help select the most relevant features from the output vector $\boldsymbol{y}$.

Algorithm 1 Locally nonlinear Explanation: GraphLIME

Input: GNN classifier f, Number of explanation features K
Input: the graph $\mathcal{G}$, the node $\boldsymbol{x}$ being explained
Output: K explanation features
 1: $\boldsymbol{X}_n = N\_hop\_neighbor\_sample(x)$
 2: $\boldsymbol{Z} \leftarrow \{\}$
 3: **for all** $x_i \in \boldsymbol{X}_n$ **do**
 4:     $y_i = f(\boldsymbol{x}_i)$
 5:     $\boldsymbol{Z} \leftarrow \boldsymbol{Z} \cup (\boldsymbol{x}_i, y_i)$
 6: **end for**
 7: $\boldsymbol{\beta} \leftarrow$ HSIC Lasso($\boldsymbol{Z}$) $\triangleright$ with x_i as features, y_i as label
 8: Select top-K features as explanations based on $\boldsymbol{\beta}$

For NHSIC($\boldsymbol{f}_k, \boldsymbol{f}_m$), if $\boldsymbol{f}_k$ and $\boldsymbol{f}_m$ are strongly dependent (i.e., redundant features), the value of NHSIC($\boldsymbol{f}_k, \boldsymbol{f}_m$) should be large and either of the two coefficients β_k and β_m tends to be zero in order to minimize the Eq. (2). This means that the redundant features will not be selected by the HSIC Lasso.

In Algorithm 1, we summarize the model-agnostic and nonlinear explanation framework based on N-hop network sampling and HSIC Lasso for GNNs as a procedure, which we call GraphLIME.

4 Experiments

In this section, we present the experiments to investigate the performance of the proposed framework GraphLIME and the existing explanation methods for GNN models using two real-world datasets. In particular, we address the following three questions: (1) Does the explainer select the real informative features from the noisy graph data, (2) Do the explanations from the explainers guide the users in ascertaining trust in the predictions, and (3) Do the explanations help further in selecting a better classifier.

4.1 Setting

We trained the GraphSAGE [10], which is a widely used GNN model, for the following explanation experiments. We performed simulated user experiments to evaluate the effectiveness of the proposed framework GraphLIME and other explanation methods. More specifically, we compared the proposed framework GraphLIME with the LIME [2] framework, which utilizes the perturbation method to sample data and train a linear explanation model lasso for selecting features as explanations according to the coefficients from the linear explanation model. In addition, we compared it with GNNexplainer [8], which utilizes mutual information to find a subgraph with associated features for interpreting GNN models[2]. We also compared it with a method based on the greedy procedure [27], which greedily removes the most contributory features of the prediction until the prediction changes, and the random procedure, which randomly selects K features as the explanations for the prediction being explained.

In our experiments, we used two graph datasets, namely Cora and Pubmed. Cora and pubmed are two publication datasets, each feature indicates the absence/presence of the corresponding word in Cora and the TF/IDF value of the corresponding word in Pubmed. More statistical details of the Cora and Pubmed datasets are given in brief in Table 1. We sampled the 2-hop network neighbors of the node being explained and then randomly split the datasets into training sets (80%) and testing sets (20%).

---

[2]https://github.com/RexYing/gnn-model-explainer

Table 1: Statistics of Cora and Pubmed Datasets.

Datasets	Cora	Pubmed
# of Classes	7	3
# of Nodes	2708	19717
# of Features	1433	500
# of Links	5429	44338

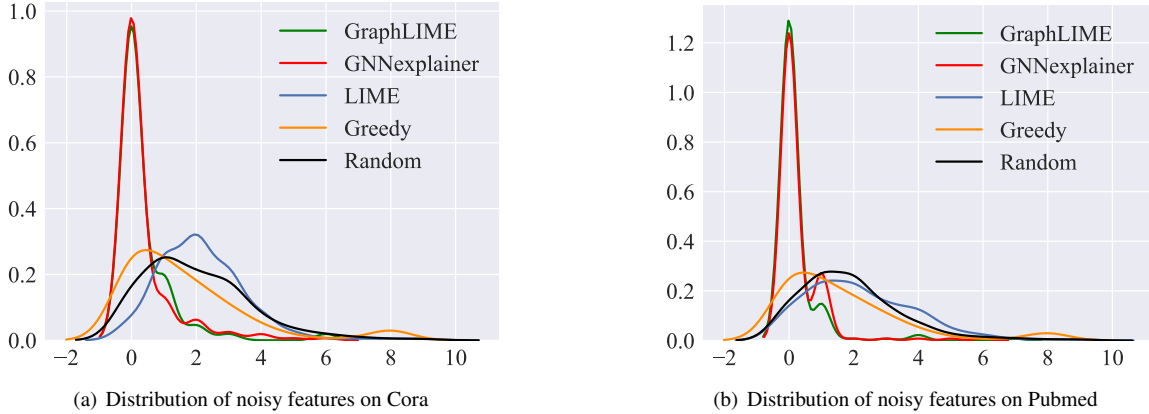(a) Distribution of noisy features on Cora   (b) Distribution of noisy features on Pubmed

Figure 2: (a) Distribution of noisy features on Cora, (b) Distribution of noisy features on Pubmed.

4.2   Does the explanation framework filter useless features?

In the first simulated user experiment, we investigated whether GraphLIME could filter useless features and select informative features as the explanations. For this, we compared the frequency of samples on different number of selected "noisy" features over different explanation frameworks to compare their abilities of denoising data.

Concretely, we artificially and randomly added 10 "noisy" features in each sample's feature vector and then trained a GraphSAGE model whose test accuracy was more than 80%. Thus we obtained the useless set of features (the 10 "noisy features") for the trained model. Finally, we produced explanations on 200 test samples for each explanation framework and compared their performance in terms of the frequency distribution of samples on different number of noisy features.

We used kernel density estimation to plot the frequency distributions of samples on different number of noisy features for five different explanation frameworks on Cora and Pubmed and used Gaussian kernel to fit the shape of distributions. The distribution is shown in Figure 2. It can be seen that the number of noisy features selected by the proposed framework GraphLIME and GNNexplainer are in general less than that selected by the other explanation frameworks. For GraphLIME and GNNexplainer, the frequency of samples on different number of selected noisy features was around 0. This means that the GraphLIME and GNNexplainer frameworks rarely select useless features as explanations, which is very useful when the graph data has a large amount of noise. LIME is not capable of ignoring the noisy features, and the distribution of LIME is mainly around 1 to 4. For Random procedure, the frequency distribution is similar to that of LIME. The Greedy explanation procedure is just slightly better than LIME and Random, but not comparable with GraphLIME and GNNexplainer. The results demonstrate that the proposed framework can denoise graph data for the GNN model when the data contains a large amount of noise and that it is more capable of finding informative features as explanations.

Table 2: Average F1-Score (%) of trustworthiness for different explainers on Cora.

Method	K=10	K=15	K=20	K=25
Random	40.4±4.3	37.7±4.6	36.6±4.3	36.0±4.2
Greedy	71.7±2.9	71.6±3.2	71.5±3.3	71.5±3.2
GNNexplainer	77.6±3.1	71.5±2.9	77.1±1.4	76.5±2.5
LIME	89.3±1.8	89.7±1.7	89.8±1.1	89.8±1.1
GraphLIME	**95.3±0.5**	**95.6±0.7**	**95.4±0.7**	**95.4±0.6**

Table 3: Average F1-Score (%) of trustworthiness for different explainers on Pubmed.

Method	K=10	K=15	K=20	K=25
Random	21.9±2.6	16.9±2.1	15.3±1.7	14.6±1.6
Greedy	63.5±7.5	62.8±7.3	62.4±7.3	62.6±7.2
GNNexplainer	79.5±2.7	77.6±2.2	79.3±3.7	75.1±1.6
LIME	83.6±0.8	84.0±0.9	84.5±0.9	84.6±0.9
GraphLIME	**92.5±0.8**	**92.3±0.8**	**91.6±0.9**	**91.5±0.8**

4.3 Do I trust this prediction?

The prediction made by a classifier model might not be credible; therefore, it is important for an explanation framework that the explanations can aid users in deciding whether a prediction is trustworthy. In this experiment, we compared this kind of ability for different explanation frameworks.

Firstly, we randomly selected 25% of the features as "untrustworthy" features. Then, we trained a GraphSAGE classifier and obtained the predictions on testing samples. We assumed that the users could identify these "untrustworthy" features and that they would not want these features as explanations. Secondly, we developed *oracle* "trustworthiness" by labeling predictions from the classifier as "untrustworthy" if the prediction changed after removing those "untrustworthy" features from the samples (because that meant the prediction was mainly decided by those "untrustworthy" features), and "trustworthy" otherwise. We regard *oracle* "trustworthiness" as the true label to decide whether a prediction is credible. Thirdly, for GraphLIME and LIME, we assumed that the simulated users regard predictions from the classifier as "untrustworthy" if the prediction made by another approximation linear model changes when all "untrustworthy" features that appear in the explanations are removed. For GNNexplainer, Greedy, and Random explainer, the prediction was deemed untrustworthy by the users if any "untrustworthy" features appeared in their explanations. Finally, we compared the decisions made by the simulated users with the *oracle* "trustworthiness".

We set the number of selected features as $K = 10, 15, 20, 25$ and reported the averaged F1-Score on the trustworthy predictions for each explanation framework over 100 rounds. The results in Table 2 and 3 show that GraphLIME is superior to the other explanation methods on the two datasets. The lower F1-Score of the other explanation methods indicate that they achieved a lower precision (i.e., trusting too many predictions) or a lower recall (i.e., mistrusting predictions more than they should), while the higher F1-Score of the proposed framework GraphLIME indicates that it achieved both high precision and high recall.
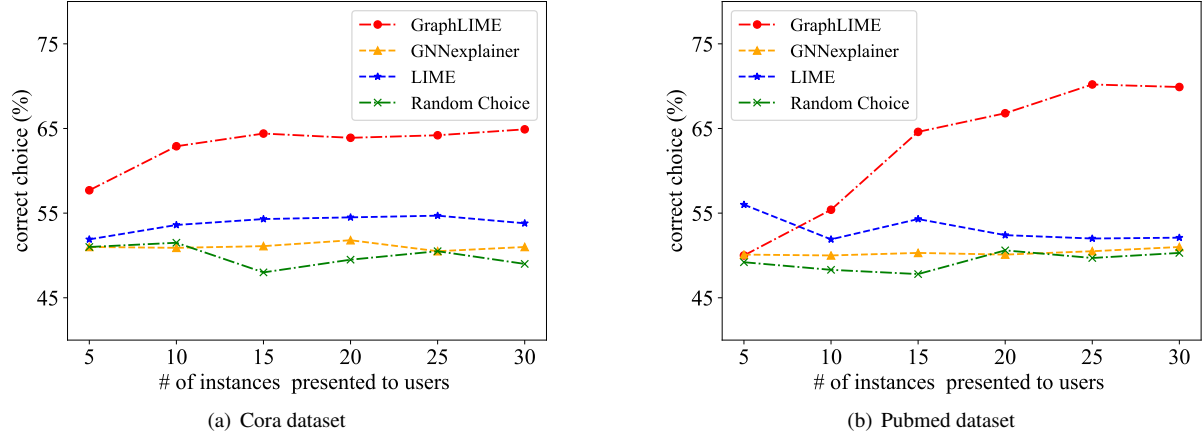
(a) Cora dataset   (b) Pubmed dataset

Figure 3: Choose the better one among the two classifiers, as the number of instances presented to users B is varied for four explanation methods. Averages over 200 rounds.

4.4 Does the explainer help to identify the better model?

In the final simulated user experiment, we explored whether the explanation framework could be used for model selection and for guiding users to choose the *better* one among two different GNN classifiers. We compared the performance of different frameworks in selecting the *better* classifier by reporting the accuracy of selecting the real better classifier.

In order to simulate a situation where the models use not only informative features in the real world but also ones that introduce spurious relations, We artificially added 10 noisy features to the data, and marked them as untrustworthy features. We created pairs of classifiers by repeatedly training pairs of GraphSAGE classifiers until their training accuracy and test accuracy were both above 80% and the difference in their test accuracy was more than 5%. Then, we obtained explanations of the samples being explained by the explanation framework on two competing classifiers and recorded the number of untrustworthy features appearing in the explanations for the two classifiers. Note that the *better* classifier should have fewer untrustworthy features in its explanations; therefore, we selected the classifier with fewer untrustworthy features as the better classifier and compared this choice with the real better classifier with higher test accuracy.

The goal of this experiment was to verify whether the explanation framework could guide users to identify better models based on explanations from the explanation framework. Considering that users may not have time to examine a large number of samples, let B donate the number of samples being explained, which they are willing to look at in order to identify the better classifier. We use Submodular Pick [2] to select B samples and present these samples to the users for examining.

We present the accuracy of selecting the real better classifier as B varies from 5 to 30, averaged over 200 rounds, in Figure 3. We compared the performance of GraphLIME, LIME, GNNexplainer, and omitted Greedy and Random because they could not produce useful explanations as shown in the previous two experiments. Moreover, we also plotted the accuracy curve for random choice. These results demonstrate that the proposed method GraphLIME can be used to perform model selection and that it outperforms LIME and GNNexplainer, which are only slightly better than the random choice. With the increase in number of presented instances, the performance of GraphLIME also improved and finally reached convergence.

5 Conclusion and Future Work

Most of the explanation methods for neural models are designed for general neural networks, while only few works exist for GNNs. In this paper, we presented a model-agnostic local interpretable explanation framework for GNN, which we

call GraphLIME. It is able to leverage the feature information of the N-hop network neighbors of the node being explained and their predicted labels in a local subgraph, utilize the Hilbert-Schmidt Independence Criterion Lasso (HSIC Lasso), which is a nonlinear interpretable model for capturing the nonlinear dependency between features and predicted outputs, and produce finite features as the explanations for a particular prediction. Experiments on two real-world graph datasets demonstrated the effectiveness of the proposed framework. It could filter noisy features and select the real informative features, guide users in ascertaining trust in predictions, and help users to identify the better classifier.

In the future, we would mainly focus on two aspects of the explanation for GNN models. Firstly, we will investigate how to extend the proposed GraphLIME so that it is able to explain the structural graph patterns and find the most important subgraph structure that influences the predictions from the GNN model. Secondly, the current framework can only explain the prediction for a single target node; therefore, we will investigate how to provide an explanation for a set of nodes, where all the nodes belong to one class from a given GNN model in a nonlinear manner.

References

[1] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Interpretable & explorable approximations of black box models. *arXiv preprint arXiv:1707.01154*, 2017.

[2] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *KDD*, pages 1135–1144. ACM, 2016.

[3] Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. Learning to explain: An information-theoretic perspective on model interpretation. *arXiv preprint arXiv:1802.07814*, 2018.

[4] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NIPS*, pages 4765–4774, 2017.

[5] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328. JMLR. org, 2017.

[6] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, pages 1885–1894. JMLR. org, 2017.

[7] Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. In *NIPS*, pages 9291–9301, 2018.

[8] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *NIPS*, pages 9240–9251, 2019.

[9] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[10] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, pages 1024–1034, 2017.

[11] Jianfei Chen, Jun Zhu, and Le Song. Stochastic training of graph convolutional networks with variance reduction. *arXiv preprint arXiv:1710.10568*, 2017.

[12] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[13] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. *arXiv preprint arXiv:1803.07294*, 2018.

[14] John Boaz Lee, Ryan Rossi, and Xiangnan Kong. Graph classification using structural attention. In *KDD*, pages 1666–1674. ACM, 2018.

[15] Chenyi Zhuang and Qiang Ma. Dual graph convolutional networks for graph-based semi-supervised classification. In *WWW*, pages 499–508. International World Wide Web Conferences Steering Committee, 2018.

[16] Jan Ruben Zilke, Eneldo Loza Mencía, and Frederik Janssen. Deepred–rule extraction from deep neural networks. In *International Conference on Discovery Science*, pages 457–473. Springer, 2016.

[17] Gregor PJ Schmitz, Chris Aldrich, and Francois S Gouws. Ann-dt: an algorithm for extraction of decision trees from artificial neural networks. *IEEE Transactions on Neural Networks*, 10(6):1392–1401, 1999.

[18] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[19] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *ICML*, pages 3145–3153. JMLR. org, 2017.

[20] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. Springer, 2014.

[21] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *NeurIPS*, pages 9505–9515, 2018.

[22] Makoto Yamada, Wittawat Jitkrittum, Leonid Sigal, Eric P Xing, and Masashi Sugiyama. High-dimensional feature selection by feature-wise kernelized lasso. *Neural Computation*, 26(1):185–207, 2014.

[23] Makoto Yamada, Jiliang Tang, Jose Lugo-Martinez, Ermin Hodzic, Raunak Shrestha, Avishek Saha, Hua Ouyang, Dawei Yin, Hiroshi Mamitsuka, Cenk Sahinalp, et al. Ultra high-dimensional nonlinear feature selection for big biological data. *TKDE*, 30(7):1352–1365, 2018.

[24] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[25] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (8):1226–1238, 2005.

[26] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *ALT*, pages 63–77. Springer, 2005.

[27] David Martens and Foster Provost. Explaining data-driven document classifications. 2013.