HENCE-X: Toward Heterogeneity-agnostic Multi-level Explainability for Deep Graph Networks

Ge Lv
HKUST
glvab@connect.ust.hk

Chen Jason Zhang
Hong Kong Polytechnic University
jason-c.zhang@polyu.edu.hk

Lei Chen
HKUST
leichen@ust.hk

ABSTRACT

Deep graph networks (DGNs) have demonstrated their outstanding effectiveness on both heterogeneous and homogeneous graphs. However their black-box nature does not allow human users to understand their working mechanisms. Recently, extensive efforts have been devoted to explaining DGNs' prediction, yet heterogeneity-agnostic multi-level explainability is still less explored. Since the two types of graphs are both irreplaceable in real-life applications, having a more general and end-to-end explainer becomes a natural and inevitable choice. In the meantime, feature-level explanation is often ignored by existing techniques, while topological-level explanation alone can be incomplete and deceptive. Thus, we propose a heterogeneity-agnostic multi-level explainer in this paper, named HENCE-X, which is a causality-guided method that can capture the non-linear dependencies of model behavior on the input using conditional probabilities. We theoretically prove that HENCE-X is guaranteed to find the Markov blanket of the explained prediction, meaning that all information that the prediction is dependent on is identified. Experiments on three real-world datasets show that HENCE-X outperforms state-of-the-art (SOTA) methods in generating faithful factual and counterfactual explanations of DGNs.

PVLDB Reference Format:
Ge Lv, Chen Jason Zhang, and Lei Chen. HENCE-X: Toward Heterogeneity-agnostic Multi-level Explainability for Deep Graph Networks. PVLDB, 14(1): XXX-XXX, 2023.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/Gori-LV/HENCE-X.

1 INTRODUCTION

Deep graph networks (DGNs) have emerged as the state-of-the-art (SOTA) techniques for graph learning owing to their ability to combine node features and graph topology. These models not only succeed on homogeneous graphs in various tasks [14, 46, 53], but also shine on heterogeneous graphs by incorporating node and edge type information [8, 12, 24, 49, 58]. However, despite their outstanding performance, DGNs still serve as back-box predictors, which hinders them from becoming trustworthy tools with transparent decision-making mechanisms. Nevertheless, understanding DGNs' behavior is of great value, as it allows end users to interpret reasoning behind predictions. Moreover, explanation of DGN's decisions helps experts examine the model and correct possible systematic errors before real-life deployment. Specifically, explaining a DGN's prediction is to answer the question "*which part of the input causes the model to make such a prediction?*". For example, consider explaining a heterogeneous DGN predicting research areas of a target scholar, *Joan F.* (filled in **orange**), based on her citation network[1] as shown in Figure 1. The input graph contains three types of nodes: *author*, *paper*, and *conference*, which are in **yellow**, **blue**, and **green**, respectively. Research keywords and words in titles are encoded using bag-of-words as features for the *author* and *paper* nodes, respectively. *Author* and *conference* nodes are labeled as one of the four areas: Data Mining (*DM*), Database (*DB*), Artificial Intelligence (*AI*) and Information Retrieval (*IR*), which is italicized in brackets. To explain the DGN's prediction "*Joan is a Database*

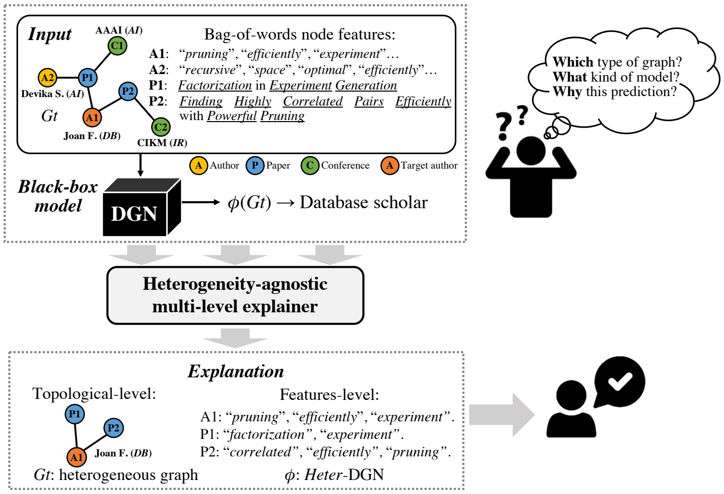

Input
AAAI (AI)
C1
A2
P1
Devika S. (AI)
P2
Gt
A1
Joan F. (DB)
C2
CIKM (IR)
Bag-of-words node features:
A1: "pruning", "efficiently", "experiment"…
A2: "recursive", "space", "optimal", "efficiently"…
P1: Factorization in Experiment Generation
P2: Finding Highly Correlated Pairs Efficiently with Powerful Pruning
A Author
P Paper
C Conference
A Target author
Black-box model
DGN
$\phi(Gt)$ → Database scholar
Which type of graph?
What kind of model?
Why this prediction?
? ?
Heterogeneity-agnostic multi-level explainer
Explanation
Topological-level:
P1
P2
A1
Joan F. (DB)
Features-level:
A1: "pruning", "efficiently", "experiment".
P1: "factorization", "experiment".
P2: "correlated", "efficiently", "pruning".
Gt: heterogeneous graph
ϕ: Heter-DGN

Figure 1: Illustrative example of explaining a heterogeneous DGN predicting research area of a scholar, *Joan F.*, based on her citation network. The input graph contains three types of nodes: *author*, *paper*, and *conference*. Features of *author* and *conference* nodes are bag-of-words of their research keywords and words in the titles, respectively. *Author* and *conference* nodes are labeled by their areas (*italicized* in brackets). (i) Explanation of the DGN's prediction "*Joan is a Database scholar*" includes a subgraph of the input graph (topological-level) and a subset of node features (feature-level) that are the most influential on the model's decision. (ii) A general and automatic explainer should provide one-stop answers to all concerns of the end user. (iii) To explain DGN's prediction, topological-level knowledge alone may be misleading, as the neighborhood does not contain information related to "DB". (iv) *A1* (the target node) should be considered as having more critical features than *A2* (2-hop neighbor); *P1* and *P2* can have different feature-level explanations as they have different feature values.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

[1] DBLP dataset.

scholar", an explainer aims to identify a compact subgraph of *Joan*'s citation network and a small subset of node features, *i.e.* the subset of research words that are most influential to the model's decision.

Although extensive efforts have been devoted to promoting the interpretability of DGNs, heterogeneity-agnostic multi-level explainability is still under shallow exploration [16, 56]. Since both homogeneous and heterogeneous graphs remain irreplaceable in practical applications, having a more general and end-to-end explainer becomes a natural and inevitable choice. A heterogeneity-agnostic explainer is highly user-friendly to non-expert practitioners, and can help all potential users to the greatest extent possible. Without requiring any domain knowledge, it offers a one-stop answer to all concerns of end users - "*Which type of graph? What model? Why this prediction?*" as shown in Figure 1. In addition, a reliable explainer should be multi-level, and include both topological- and feature- levels, to fully explore the knowledge enclosed in the graph data. However, feature-level explanation is ignored by most existing methods [21, 23, 42, 54, 57], which can lead to unreasonable or even misleading explanations. Recall the example in Figure 1, to explain DGN's predicted label for *Joan*, *i.e.* Database (*DB*), a topological-level explainer can only investigate relational information in the graph structure: "*Joan published a paper at CIKM (IR conference); she also collaborated with an AI researcher and had one publication at AAAI (AI conference)*". Evidently, topological-level information alone is incomplete and deceptive, meaning that a topology-only explainer may cause the model to incorrectly deceptive and waste resources. While feature-level explanation can explore the bag-of-words features and observe that keywords of the target author include representative ones for the DB area such as "*pruning*" and "*efficiently*"; moreover, her published papers have largely exploited DB techniques such as "*factorization*" and "*pruning*".

In a nutshell, DGN explainers should be both heterogeneity-agnostic and multi-level to ensure adaptability and reliability. Yet existing studies fall short in maintaining both capabilities. The majority of current explainers remain topological-level [3, 21, 23, 42, 51, 52, 57], which cannot be easily extended to multi-level as they are oriented towards the combinatorial nature of graph topology. While the issue with existing multi-level methods [45, 54] is that they assume all nodes share a uniform feature-level explanation. Firstly, this assumption is not appropriate for heterogeneous graphs, as nodes of different types have different feature spaces (*e.g.*, one-hot encoding v.s. bag-of-words). Secondly, nodes with different structural positions or realizations of features do not share the same explanation. In *Joan*'s example (Figure 1), ***A1*** may have more critical features than ***A2***, as ***A1*** is the target node itself, while ***A2*** is a 2-hop neighbor; ***P1*** and ***P2*** are both 1-hop neighbors of the target, yet they have totally different features. Hence the four nodes should be considered as having exclusive explanations. ***By and large***, recent techniques have taken one of two primary routes: metric-based and causality-guided. The former line of works [3, 21, 23, 42, 45, 51, 52, 54, 57] adopts heuristic metrics to quantify "*explainability*" for designing optimization objectives or loss functions. Although they justify their proposed metrics according to various theories (*e.g.*, Mutual Information and Shapley values), there still lacks of a comprehensive and object unified standard. The causality-guided approach, on the other hand, can avoid computing human-defined "explainability"; instead, causal interaction between input elements and the model's prediction is investigated to find the true cause of the target prediction. Thus, Vu and Thai [47] propose PGM-Explainer - so far the only SOTA causality-guided method. In PGM-Explainer, a Bayesian network (BN) is adopted to model the non-linear dependency of model's prediction on the input. Yet, it fails to preserve the connectivity of selected important nodes, which violates the message passing principle of DGNs.

In this work, we aim for a heterogeneity-agnostic multi-level explainer for general DGNs. To address the discussed shortcomings, one needs to develop a causality-guided, *i.e.*, BN-based explainer that ensures the connectivity of the output subgraph for a DGN, which can be either homogeneous or heterogeneous. Furthermore, the explainer should be multi-level and provide feature-level explanation for nodes in the topological-level explanation. The problem is challenging for the following three reasons: first, aiming for two levels of explanation, the learned Bayesian network should encode the ontological subordination of features to the nodes that carry them, yet enforcing such relations into the causal network will post too strong assumptions on the existence of a perfect map[2] over the distribution of input-model interaction; second, a Bayesian network that is capable of measuring causal effects of each feature on different nodes individually can possibly be very large and exceedingly expensive to learn; third, the desired Bayesian network structure does not follow the input graph topology, thus developing an effective and efficient BN learning algorithm that preserves connectivity of the output subgraph is non-trivial.

Aiming at tackling the above issues, in this paper we propose a **H**eterogen**E**ity-ag**N**osti**C** multi-l**E**vel DGN explainer, named HENCE-X. In HENCE-X, individual features are modeled as random variables while nodes are not, such a strategy allows an arbitrary model behavior distribution to have a perfect map as it posts no requirements for representing the affiliation of features to nodes; HENCE-X then learns a surrogate Bayesian network to capture the relations between features and nodes as well as the dependency of model's prediction on the input. We introduce an efficient BN structure learning algorithm based on graph traversal to find a subset of features that the prediction is dependent on, meanwhile, nodes carrying the selected feature are guaranteed to be connected. In this way, our method produces faithful and multi-level explanations for DGNs in an integrated fashion. We theoretically prove that the proposed explainer outputs a Markov blanket of the model prediction, which means HENCE-X can always locate all variables that determine the model's behavior.

2 BACKGROUND AND DEFINITIONS

In this work, we target a model-agnostic explainer, which means it does not rely on any internal information of the pretrained DGN. The explainer is allowed to perform multiple queries to the model, yet access to knowledge inside the black-box DGN such as parameters and gradients is forbidden. The reason being that model-agnostic explainers enjoy wider usage because DGNs are usually black-boxes and their internal structure is inaccessible in many real-life scenarios. In the following, we present preliminaries, including a description of DGN models, using a Bayesian network

[2] A Bayesian network is a *perfect map* of a distribution P if it is both an I-map and a D-map of P; formal definitions are provided in the Supplementary Materials [1].

as an interpretable model and a Markov blanket as the criterion of explanations. Finally, we formulate the problem studied in this work.

2.1 Explaining a Deep Graph Network

Without loss of generality, we denote the input graph of a DGN to be explained as $G = (\mathcal{V}, \mathcal{E}, \mathcal{X}, \zeta, \psi)$, where $\mathcal{V}$ and $\mathcal{E}$ are the set of nodes and edges in the graph. Each v is associated with a type $\zeta(v)$, and each edge e also has a type $\psi(e)$. Nodes are associated with features $\mathcal{X} = \{x_{v_1}, ..., x_{v_i}\}, x_{v_i} \in \mathbb{R}^{d_{\zeta(v_i)}}$, where $d_{\zeta(\cdot)}$ denotes feature dimensions of a node type. If the model to be explained is a homogeneous DGN, ζ and ψ have only one type in their range; otherwise the model is a heterogeneous DGN, ζ and ψ will map nodes and edges to different types, such that input node features may have different dimensions. Furthermore, we denote $\phi : G_t \mapsto \{1, ..., C\}$ as the pretrained DGN that maps a target instance t to one of the C classes, where G_t is the computational graph of t.

2.2 A Bayesian Network as Interpretable Model and Markov Blanket as Criterion

To avoid relying on the linear-independence assumption of the input, the Bayesian network [32] is introduced to the DGN explanation task by Vu and Thai [47]. The advantage is two-fold: first, the Bayesian network as a graph-based model allows direct and natural adaption to explaining DGNs by treating nodes in the input graph as random variables in the Bayesian network; second, it is capable of encoding complex distributions in a multi-dimensional space while providing intuitive interpretation of the dependencies using probabilistic graphical model. Specifically, nodes in the input graph are modeled as random variables, which take binary values to record whether the underlying node is perturbed. The DGN prediction is also modeled as a random variable that indicates whether the prediction has changed due to the perturbation of input. In such matters, a model's behavior is encoded into a probability distribution with multiple random variables, and we term this distribution *model behavior distribution.* Assume there exists a perfect map $\mathcal{B}^*$ of the underlying distribution, structure learning algorithm for a Bayesian network can then be employed to find the network structure of $\mathcal{B}^*$ [13, 15]. In this way, causality between a DGN's decision and the input can be unveiled as dependencies of the prediction variable on the input node variables, and the learned Bayesian network precisely visualizes the model's decision making mechanism. For ease of illustration, we simply use *variable* to refer to *random variable* when the context is clear.

As a matter of fact, understanding a DGN prediction is not necessarily of concern to the entire Bayesian network over the model behavior space; instead, only the critical parts of the input are of interest because end-users prefer a simple and human-intelligible explanation. In addition, learning the entire network can be redundant and possibly cumbersome when the input space is large. Hence a Markov blanket in a Bayesian network is further introduced as the criterion of explanations [47]. The term was coined by Judea Pearl [33] to study probabilistic reasoning and plausible inference; the formal definition is shown in below:

Definition 2.1 (Markov blanket[33]). *Consider a joint distribution on a set of random variables $\{Y\} \cup Z$, where $Z = \{X_1, X_2, ...X_n\}$. A Markov blanket of Y, denoted by $\mathrm{MB}(Y)$, is a subset of Z, conditioned on which Y is independent of any other variables in Z, i.e.,*

$$Y \perp\!\!\!\perp_{\mathcal{B}} Z \backslash \mathrm{MB}(Y) \Big| \mathrm{MB}(Y).$$

In a Bayesian network, the Markov blanket of a variable Y includes its parents, children and spouses, such that the blanket d-separates[3] Y from all other nodes in Z. By definition, a Markov blanket contains all dependencies of the target variable in the distribution. In our problem, under the assumption that there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution, a Markov blanket of the prediction variable, denoted by $\mathrm{MB}_{\mathcal{B}^*}(\Phi_t)$, encloses all the statistical information that determines the prediction and changes in the model's decision are independent of any variable outside the blanket. Hence, the Markov blanket serves as a duteous criterion for precise and concise explanations.

2.3 Towards Multi-level Explanation

Despite the advantage of not being based on a linearly independent assumption on input elements and the sophistication of employing a Markov blanket to determine an explanation, modeling nodes in the input graph as random variables suffers from an intrinsic drawback: it falls short of generating feature explanations as a matter of course. Towards heterogeneity-agnostic and multi-level explainability, we propose using each entry of node features instead of the node as random variables, such that the explainer can capture node-specific feature-level importance. In this way, searching for variables in the Markov blanket over a mapped Bayesian network changes from finding a subset of nodes in the input graph to identifying a subset of entries in the feature matrix. In the first place, we formally define the *model behavior distribution* studied in this paper as below:

Definition 2.2 (Model behavior distribution). *Given a pretrained DGN ϕ and a target node t, denote j-th feature on node v_i as $f_{v_i}^{(j)}$ for all nodes in t's computational graph G_t. Let each $f_{v_i}^{(j)}$ in the input feature matrix $\mathcal{X}_{G_t}$ be associated with a random variable $F_{v_i}^{(j)}$ to encode whether the underlying feature is perturbed, and let the DGN's prediction on t also be associated with a random variable Φ_t to encode the prediction change when the perturbed input is fed into the DGN. The model behavior distribution of ϕ on t, denoted by $\mathcal{P}_\phi(t)$, is defined as the probability distribution formed by perturbing $\mathcal{X}_{G_t}$ and the induced DGN prediction change.*

For simplicity of notation, we denote the set of all variables associated to features in the distribution as $\mathcal{F}_\phi(t)$. DGN prediction is modeled as an individual random variable and the model's behavior does not affect the input in return, hence the prediction variable has *no child* in the distribution. Model behavior distribution allows one to explore the complex causality between input elements and the model's decision by revealing reasoning patterns in a Bayesian network of their interaction. Denote $d_{\zeta(\cdot)}^*$ as the maximum feature dimension among all nodes types, we now formally define the problem studied in this work as below:

Definition 2.3 (Heterogeneity-agnostic multi-level explanation generation problem). *Given a pretrained DGN ϕ and a*

[3] Two nodes X and Y in a Bayesian network are *d-separated* by a set W if all paths between X and Y are blocked by W; equivalently, $X \perp\!\!\!\perp Y \Big| W$. Formal definition of *d-separation* is provided in the Supplementary Materials [1].

target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_\phi(t)$, the problem of generating heterogeneity-agnostic multi-level explanation for the model's prediction on t is to identify a subset of entries in the input feature matrix X_{G_t}, denoted by $\mathcal{E}_{feat}$, such that $\mathcal{M} = \left\{F_{v_i}^{(j)} : \forall f_{v_i}^{(j)} \in \mathcal{E}_{feat}\right\}$ is a Markov blanket of Φ_t in $\mathcal{B}^*$ and the subgraph of G_t induced by $\mathcal{E}_{topo} = \left\{v_i : \exists j \in [0..d^*_{\zeta(\cdot)}] \text{ s.t. } f_{v_i}^{(j)} \in \mathcal{E}_{feat}, \forall v_i \text{ in } G_t\right\}$ is a connected component.

3 THE PROPOSED EXPLAINER

Given a pretrained DGN and a target node t, the goal of HENCE-X is to identify a subset of features on nodes in t's computational graph that are crucial to the model's prediction on t. To achieve this goal, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_\phi(t)$, the Markov blanket of the prediction variable can be learned by first collecting a group of samples from the underlying distribution, then the blanket is searched based on the observed data [15, 25, 40]. The proposed HENCE-X consists of three principal components: sample generation, Markov blanket searching and network parameter learning. The sample generation step is to collect a set of realizations of variables in $\mathcal{P}_\phi(t)$ as sampled data. Next, the Markov blanket searching step is designed for identifying a subset of $\mathcal{F}_\phi(t)$ that forms a Markov blanket of the prediction variable based on the collected samples. Finally, we introduce the parameter learning step to quantify exact contribution of each element in the searched blanket and further produce factual and counterfactual explanations of the prediction. Each of them are detailed in the following sections.

3.1 Sample Generation

Aiming for a group of samples drawn from the model behavior distribution, we perturb the feature matrix of the target while the corresponding prediction scores are queried from the pretrained DGN. Different from the existing BN-based explainer [47], HENCE-X perturbs every entry in the feature matrix severally instead of conducting node-wise perturbation, such that the importance of each feature can be measured individually but not collectively. In particular, the realization rule of feature random variables is formulated as below:

$$F_{v_i}^{(j)} = \begin{cases} 1, & \text{if } f_{v_i}^{(j)} \text{ is perturbed;} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The perturbation scheme must be carefully designed in order to properly eliminate the effect of information provided by the underlying feature. Such action should be determined by both the nature of features in the data and the application scenario of the DGN to avoid constructing illegal inputs.

After perturbing the features and acquiring a varied input G'_t, the corresponding DGN behavior, *i.e.*, a new prediction score $\phi(G'_t)$, can be obtained by feeding the modified input to the model. To align with the input variables, the existing Bayesian-network-based explainer [47] casts the continuous score into a categorical random variable by indicating if the new score is lower than the original score by more than a threshold. This strategy is deficient for two reasons. First, it does not investigate the case where noise in the input is eliminated by perturbation, thus the DGN can be more confident about its decision and increase the prediction score. Second, it cannot capture the optimized situation where the model is most confident about its prediction, namely, when the prediction score is minimized or maximized. Thus, we introduce a new realization rule for the prediction variable Φ_t as below:

$$\Phi_t = \begin{cases} min, & \text{if } \phi(G'_t)[\hat{y} = \hat{y}_0] \leq \min_{G''_t} \phi(G''_t)[\hat{y} = \hat{y}_0] + \sigma; \\ max, & \text{if } \phi(G'_t)[\hat{y} = \hat{y}_0] \geq \max_{G''_t} \phi(G''_t)[\hat{y} = \hat{y}_0] - \sigma; \\ othw., & \text{otherwise.} \end{cases} \tag{2}$$

where $\phi(\cdot)[\hat{y} = \hat{y}_0]$ denotes that the score is specified for the original predicted label $\hat{y}_0$ on the target instance. G'_t and G''_t are perturbed graph induced by the corresponding realization of feature variables, and σ is a small constant. Here, max$(\cdot)$ and min$(\cdot)$ are measured over all predictions produced in the sample generation step. As a result, Φ_t is a categorical random variable with domain $\{min, max, othw.\}$. The realization rule is specifically designed to capture the model's behavior in respect of the two forms of explanations, *factual* and *counterfactual*: when $\Phi_t = max$ (*w.r.t. min*), the situation corresponds to a factual (*w.r.t.* counterfactual) explanation by showing that "*given current information, the DGN is most certain that the instance belongs (w.r.t. does not belong) to class $\hat{y}_0$*".

Mathematically, a sample is one specific realization of all random variables in the distribution $\mathcal{P}_\phi(t)$ according to the realization rules Equation (1) and Equation (2), which is represented as a row of categorical values and inserted into the sample table $\mathcal{D}_{P_\phi(t)}$. In such a manner, each perturbation run and model prediction query will produce one input-output pair as one sample that describes the model's behavior responding to the input variation. Upon acquisition of a sufficient number of samples, Markov blanket and parameters of the Bayesian network are then learned based on the collected data, which is detailed in the following sections.

3.2 Markov Blanket Searching

We employ the Markov blanket as the criterion for explanations as it is a subtle distinction between decisive and irrelevant factors. Under the assumption that there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution, the prediction variable is independent of all other variables conditioned on the Markov blanket. Thus the statistical information on the prediction variable embraced in $\text{MB}_{\mathcal{B}^*}(\Phi_t)$ is the same as that in $\mathcal{B}^*$ [47]. Given a sufficient number of samples $\mathcal{D}$, the Markov blanket of a target variable can be discovered by the Grow-Shrink (GS) algorithm [25], which retains *provable* correctness. In the growing phase, the algorithm starts with an empty blanket and scans all random variables in the Bayesian network, if there exists one that is not independent from the target conditioned on the current blanket, it will be added to the blanket. In the shrinking phase, the blanket is shrunk by removing the variables which the target is independent of conditioned on the others in the blanket. The algorithm ensures the property of a Markov blanket is well guarded, yet it has the exponential computational cost problem due to the conditional independence tests involved. Unfortunately, the statistical independence test conditioned on m binary variables requires 2^m *marginal* independence tests, the overall complexity becomes $O(2^m \cdot |\mathcal{D}|)$, where $|\mathcal{D}|$ is the sample size.

Consider a multi-dimensional distribution over a set of random variables Z, growing a Markov blanket requires $|Z| - 1$ conditional independence tests with growing conditioning set as the blanket expands. Suppose one is interested in finding the Markov blanket of a variable in Z which is rather small compared to $|Z|$. In the worst case, variables that belong to the Markov blanket are tested lastly, and before that each variable tested is added to the current set, then the worst-case complexity of growing the blanket is $O(\sum_{m=1}^{|Z|} 2^m \cdot |\mathcal{D}|) = O(2^{|Z|} \cdot |\mathcal{D}|)$. On the other hand, when the variables in the Markov blanket are tested at the initial steps, all the other variables will not be added to the blanket, hence the conditioning set will not grow and the best-case complexity is $O(|Z| \cdot |\mathcal{D}|)$. However, this is not realistic because searching for the Markov blanket given known variables in it is a paradox. Nevertheless, there still exists opportunities for a better solution in growing a Markov blanket: one can first screen out variables that are certainly not in the blanket; then a superb order to conduct conditional independence test on the variables can be developed to lead the algorithm to its best-case complexity; furthermore, a stop condition is also useful, namely, knowing all the untested variables are surely not in the blanket. Subsequently, number of conditional independence tests in the shrinking phase will also be reduced. With respect to these opportunities, we design corresponding modules in HENCE-X to search for a Markov blanket of the prediction variable effectively and efficiently, which are introduced in the rest of this subsection.

3.2.1 *Variable Screening.*

In contrast to the conditional independence test, the marginal independence test has a fair complexity of $O(|\mathcal{D}|)$, where $|\mathcal{D}|$ is the sample size. To borrow from this advantage, we present a theorem proposed by Vu and Thai (2020) [47] using symbols in this paper, which is equivalent to the original one.

THEOREM 3.1 ([47] THEOREM 2). *Assume there exists a perfect map $\mathcal{B}^*$ of a distribution P on a set of random variables Z. If a node $X \in Z$ has no child in $\mathcal{B}^*$, $\mathrm{MB}_{\mathcal{B}^*} \subseteq U(X)$ where $U(X) = S(X) \triangleq \{X' \in Z : X' \not\perp\!\!\!\perp_{\mathcal{B}^*} X\}$.*

The theorem describes the relation between the Markov blanket of a target random variable with no child and the set of other variables built using marginal independence tests. According to this theorem, we have the following lemma that serves as a guide for screening out unimportant variables.

PROPOSITION 3.2. *Given a pretrained DGN ϕ and a target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_\phi(t)$, let*

$$\mathcal{K}_{\Phi_t} = \left\{F_{v'}^{(j')} \in \mathcal{F}_\phi(t) : \Phi_t \not\perp\!\!\!\perp_{\mathcal{B}^*} F_{v'}^{(j')}\right\}. \tag{3}$$

For any variable $F_{v_i}^{(j)} \in \mathcal{F}_\phi(t)$, if $F_{v_i}^{(j)} \notin \mathcal{K}_{\Phi_t}$, then $F_{v_i}^{(j)} \notin \mathrm{MB}_{\mathcal{B}^*}(\Phi_t)$.

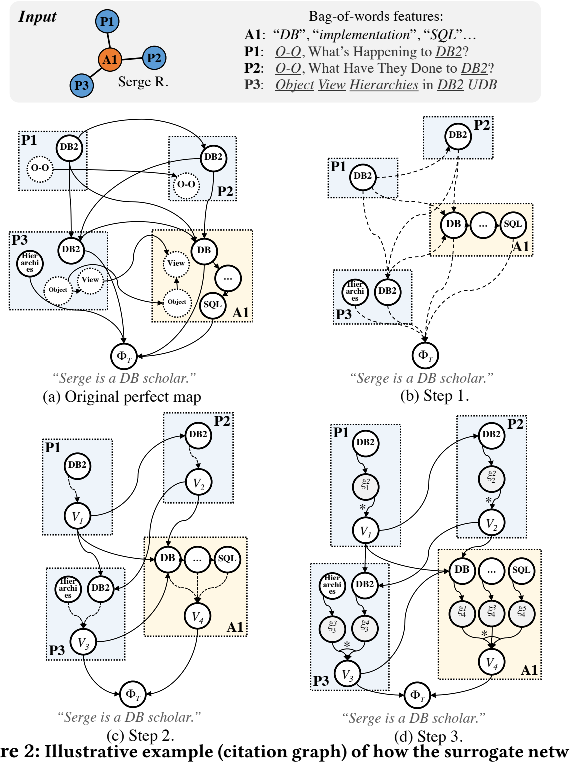
The proof directly follows from Theorem 3.1 as marginal independence is commutative. We term $\mathcal{K}_{\Phi_t}$ the *blanket basket* of the prediction variable, because $\mathcal{K}_{\Phi_t}$ plays the role of a container that holds the blanket, and the blanket never oversteps the capacity of $\mathcal{K}_{\Phi_t}$. Conversely, $\mathcal{K}_{\Phi_t}$ can be much larger than $\mathrm{MB}_{\mathcal{B}^*}(\Phi_t)$, especially when the ancestral set of the target in the Bayesian network is large. As a matter of fact, the existing Bayesian-network-based explainer [47] simply takes the top-k marginally dependent variables in the blanket basket as the output explanation. This is problematic as the prediction may retain high marginal dependency on a variable V' due to V' being a significant non-parent ancestor of the prediction, but conditioned on the direct parent(s), the prediction is independent of V'. Hence taking only the top-k marginally dependent variables may lead to an incomplete Markov blanket.

To give an example of blanket basket, in Figure 2 shows an illustrative instance using a snapshot of the DBLP citation network, the original input is presented at the top. The underlying perfect map $\mathcal{B}^*$ of the model behavior distribution is shown in subplot (a), where edges are drawn in *curves* rather than straight lines to distinguish between Bayesian network and the input graph; random variables are represented by white nodes, while those not in the blanket basket are drawn using dotted line. The paper **P1**"*O-O, What's Happening to DB2?*" has the word "*O-O*" in its features, which is the cause of the same word in the later paper **P2** "*O-O, What Have They Done to DB2?*". However, "*O-O*" is not connected to the prediction variable, which is then marginally independent of "*O-O*", hence it is not included in the blanket basket.

Blanket Basket Searching for Screening Variables. A cannier approach to making use of the blanket basket is to trim variables that are not necessary for the Markov blanket, such that unnecessary conditional independence tests on those variables are avoided when growing the blanket. Formally, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution, the variable screening step is to search for the blanket basket of the prediction variable Φ_t by conducting marginal independence tests between Φ_t and every feature variable $F_{v_i}^{(j)} \in \mathcal{F}_\phi(t)$, then prune all $F_{v_i}^{(j)}$'s that Φ_t is marginally independent of; only those in the blanket basket will proceed to the next step. Below we detail how to model the causal effect of vertices in the input graph on the model's predictions.

3.2.2 *Surrogate Bayesian Network.*

Though aiming at multi-level explanations, HENCE-X does not design vertex variables together with feature variables, nor does it further enforce the association of feature variables with vertex variables. The reason for this is it cannot be guaranteed that a Bayesian network with certain human-defined local structures is indeed a perfect map of the model behavior distribution. Thus, HENCE-X introduces feature and prediction variables only to allow an arbitrary-structured perfect map to exist. However, such Bayesian networks ignore the fact that individual features on the same vertex are processed by the DGN as an entirety and transformed into neural message in union, thus they have impact on the prediction in an integrated fashion. To bridge this gap, we build a surrogate Bayesian network $\mathcal{B}'$ based on the original perfect map $\mathcal{B}^*$, in which synthetic nodes are added as convergent variables for vertexes in the input graph. By dealing with vertex variables, HENCE-X can capture the integrated causal effects of features on the same vertex and better explore the graph topology to discover critical message passing paths. Furthermore, guided by the graph topology, an efficient and effective Markov blanket searching algorithm can be developed. In the following, we describe using steps on how to build the surrogate network and theoretically prove that the Markov blanket of the prediction variable Φ_t in the original perfect map $\mathcal{B}^*$ can be successfully acquired from one in the surrogate network $\mathcal{B}'$. To avoid ambiguity, we use *vertex* to refer to the node in the input graph for the next subsection.


Input
P1
P2
P3
A1
Serge R.
Bag-of-words features:
A1: "DB","implementation","SQL"…
P1: O-O, What's Happening to DB2?
P2: O-O, What Have They Done to DB2?
P3: Object View Hierarchies in DB2 UDB
DB2
O-O
View
Object
Hier archi es
DB
...
SQL
Φ_t
V_1
V_2
V_3
V_4
*
"Serge is a DB scholar."

(a) Original perfect map

(b) Step 1.

(c) Step 2.

(d) Step 3.

Figure 2: Illustrative example (citation graph) of how the surrogate network $\mathcal{B}'$ is built based on the original perfect map $\mathcal{B}^*$; curved edges are used to distinguish between BN and the input graph. (a) The perfect map $\mathcal{B}^*$. (b) *Step 1.* Introducing variables in the Blanket Basket to $\mathcal{B}'$; dotted edges are the original ones in $\mathcal{B}^*$. (c) *Step 2.* Introducing synthetic variable V_i's for vertexes in the input graph and network structure transfer. (d) *Step 3.* Introduce contribution variable ξ_i^j's and the realization rule of synthetic variables, then the surrogate network $\mathcal{B}'$ is completed.

Step 1. Introducing variables in the blanket basket to $\mathcal{B}'$. According to Proposition 3.2, the desired Markov blanket must be a subset of variables in the blanket basket $\mathcal{K}_{\Phi_t}$. Hence we introduce all variables in $\mathcal{K}_{\Phi_t}$ and the prediction variable Φ_t to the surrogate network. The blanket basket is analog to the *causal basin* defined for poly-trees, which contains causal flows for recovering causality patterns from statistical data [19, 30, 31, 37, 43, 48]. Likewise, the blanket basket also contains all causes of the model's prediction, thus the constructed surrogate networks must include all its variables. This step is shown in Figure 2(b), feature variables such as "*DB2*" from **P1**, **P2** and **P3**, in addition to "*DB*" and "*SQL*" from **A1** are introduced to $\mathcal{B}'$, but "*O-O*" from **P1** and **P2** is not. Dotted lines in the subplot represent the induced substructure of $\mathcal{B}^*$. As illustrated, all the ancestors (causes) of Φ_t are included in $\mathcal{B}'$.

Step 2. Introducing synthetic variables V_i's for vertexes and network structure transfer from $\mathcal{B}^*$ to $\mathcal{B}'$. To model the fact that features on the same vertex are processed by the DGN as an entirety, we introduce a synthetic variable for each vertex with feature(s) in the blanket basket, such that the causal effects of its features can be captured in an integrated fashion. This is then represented by transferring structures of $\mathcal{B}^*$ to $\mathcal{B}'$. Denote the set of related vertexes with respect to the blanket basket as

$$\mathcal{V}_\mathcal{K} = \left\{v_i : \exists j \in [0..d_{\zeta(v_i)}] \ s.t. \ F^{(j)}_{v_i} \in \mathcal{K}_{\Phi_t}, \ for\ all\ v_i\ in\ G_t\right\}, \quad (4)$$

and the set of feature variables in the blanket basket that are on the same vertex v_i as

$$\mathcal{K}_{\Phi_t}(v_i) = \left\{F^{(j)}_{v'} \in \mathcal{K}_{\Phi_t} : v' = v_i, j \in [0..d_{\zeta(v_i)}]\right\}. \quad (5)$$

For each vertex v_i in $\mathcal{V}_\mathcal{K}$, a *synthetic variable* V_i [29] is added to $\mathcal{B}'$ to aggregate the outgoing causative impact of feature variables in $\mathcal{K}_{\Phi_t}(v_i)$. Next, the structure of $\mathcal{B}^*$ is transferred to $\mathcal{B}'$ in order to ensure the latter contains the same causal flow as the former. Specifically, for each $v_i \in \mathcal{V}_\mathcal{K}$, if there exists an edge in $\mathcal{B}^*$ outgoing from a variable in $\mathcal{K}_{\Phi_t}(v_i)$ to another variable in $\{\Phi_t\} \bigcup \mathcal{K}_{\Phi_t} \Big\backslash \mathcal{K}_{\Phi_t}(v_i)$, add an edge into $\mathcal{B}'$ from V_i to the latter variable. In this way, the causal effects of feature variables on a vertex v_i to their children are aggregated by V_i. Figure 2(c) illustrates this step, V_1, V_2, V_3 and V_4 are introduced for **P1**, **P2**, **P3** and **A1**, respectively. Take **P3** as an example, impacts of the features "*Hierarchies*" and "*DB2*" on model prediction are integrated by V_3 after the structure transfer. Synthetic variable was coined by Neil, *et al.* [29] for combining multiple variables in local structures of large-scale Bayesian networks, which has been proven to be effective for reducing combinatorial explorations in many areas such as software engineering [28] and medical decision support [5]. We adopt synthetic variables here to mimic the message passing scheme of DGN: each feature has an impact on the model's prediction to varying degrees, yet features on the same vertex function as an entirety. Moreover, introducing vertex representation allows the explainer to explore the graph topology. The next step describes how the value of a synthetic variable is determined by the corresponding feature variables.

Step 3. Introduce contribution variable ξ_i^j's and realization rule of synthetic variables. Before deciding the realization rule of the synthetic variables, it is noteworthy that, in the BN that encodes the non-linear dependency of model behavior on the input, explaining a prediction does not concern how features on the same vertex affect each other. Instead, only causal effects of input elements on the prediction variable are of interest. Hence, to the simplify local structure of variables associated with a vertex and its features, in addition to determining the realization of V_i, we introduce the convergent variable in causal independence [44, 59] to capture how $F^{(\cdot)}_{v_i}$ collectively affect V_i. Specifically, for each vertex v_i in $\mathcal{V}_\mathcal{K}$, we introduce for each variable $F^{(j)}_{v_i}$ in $\mathcal{K}_{\Phi_t}(v_i)$ a *contribution variable* ξ_i^j. Each ξ_i^j has only one incoming edge from $F^{(j)}_{v_i}$ and only one outgoing edge to V_i. As a result, each piece of the BN fragments (local structures) consists of V_i, $F^{(\cdot)}_{v_i}$ and ξ_i^j's corresponding to one vertex in the input graph, and the synthetic variable is dependent on its parent contribution variables.

By definition, for any vertex v_i in $\mathcal{V}_\mathcal{K}$, all variables in $\mathcal{K}_{\Phi_t}(v_i)$ are *causally independent* [59] with respect to V_i as there exist random variable ξ_i^j's such that $\xi_i^{j'}$ is probabilistically dependent on $F^{(j')}_{v_i}$ *only*, and it is conditionally independent of all other variables in $\mathcal{K}_{\Phi_t}(v_i)$ and all other ξ_i^j's given $F^{(j')}_{v_i}$; and there exists some commutative and associative binary operator $*$ such that

$$V_i = \xi_i^j * \xi_i^{j'} * \ldots * \xi_i^{j''}, \forall F^{(j)}_{v_i} \in \mathcal{K}_{\Phi_t}(v_i).$$

Put simply, individual contributions from different causes can be considered independent and the total influence on V_i is a combination of those individual ones. Naturally, we let ξ_i^j be the presence of the underlying feature and the realization rule is shown below:

$$\xi_i^j = 1 - F_{v_i}^{(j)}.$$

Using a vector to represent all variables in $\mathcal{K}_{\Phi_t}(v_i)$ and their corresponding ξ_i^j, it can be rewritten as

$$\boldsymbol{\xi}_i = \mathbf{1} - \mathbf{F}_{v_i}.$$

Furthermore, Noisy-AND gate [59] is introduced to be the commutative and associative binary operator $*$, namely, V_i is determined by the situation ξ_i^j <u>and</u> $\xi_i^{j'}$ present. For example, if the research word "*knowledge graph*" presents in the bag-of-words features, only when "*knowledge*" <u>and</u> "*graph*" present simultaneously, the information is complete, as "*knowledge*" or "*graph*" alone will lead to a very different meaning. Take **A1** in Figure 2(d) as an example, "DB", "SQL" and other features that are associated with **A1** are said to be *causally independent* with respect to V_4, if there exist random variables ξ_4^j's such that, every $\xi_4^{j'}$ is probabilistically dependent on $F_{v_4}^{(j')}$ only and is conditionally independent of all other variables in $\mathcal{K}_{\Phi_t}(v_4)$ and all other ξ_4^j's given $F_{v_4}^{(j')}$.

Now we are ready to introduce the realization rule of V_i. As Noisy-AND gate is used as the binary operator, as a matter of fact, $\boldsymbol{\xi}_i$ directly encodes the case of AND by recording the presence of the underlying features. To align with the prediction variable and feature variables that are all categorical random variables, the random vector should be also transformed into a categorical variable. Formally, realization rule for synthetic variables is shown below:

$$V_i = categorize(\boldsymbol{\xi}_i),$$

where $categorize(\cdot)$ can be any one-to-one function that maps a vector into a unique categorical value. In this paper, we simply use a dot product function

$$g(\boldsymbol{x}) = \boldsymbol{x} \cdot [c^0, c^1, ..., c^{|\boldsymbol{x}|}],$$

for any integer $c > 1$, such that the input random vector can be transformed into an integer-valued categorical random variable. It can be easily verified that, as long as the one-to-one mapping is well maintained, the transformed variable and the random vector are equivalent by proving they have the same probability density function. In this way, vertexes in the computational graph are also represented by categorical random variables and statistical independence of the prediction variable on vertexes can be computed.

3.2.3 Relation between $\mathcal{B}^*$ and $\mathcal{B}'$. The surrogate network $\mathcal{B}'$ is built to facilitate finding the Markov blanket $\text{MB}_{\mathcal{B}^*}(\Phi_t)$ in the perfect map $\mathcal{B}^*$, which is specially designed in a way that $\text{MB}_{\mathcal{B}^*}(\Phi_t)$ can be guaranteed form a Markov blanket $\text{MB}_{\mathcal{B}'}(\Phi_t)$ in $\mathcal{B}'$. Before we go to the algorithm, we present the relation between the perfect map and the surrogate network in this subsection. The below lemma states that one can successfully acquire $\text{MB}_{\mathcal{B}^*}(\Phi_t)$ from $\text{MB}_{\mathcal{B}'}(\Phi_t)$ under a certain condition.

Lemma 3.3. *Let $\text{MB}_{\mathcal{B}'}(\Phi_t)$ be a Markov blanket of Φ_t in the surrogate network $\mathcal{B}'$ that does not contain any contribution variables, denote $\mathcal{M}$ as the subset of $\text{MB}_{\mathcal{B}'}(\Phi_t)$, which includes all feature variables only; mathematically,*

$$\mathcal{M} = \left\{F_{v_i}^{(j)} : \exists j \in [0..d_{\zeta(v_i)}] \text{ s.t. } F_{v_i}^{(j)} \in \text{MB}_{\mathcal{B}'}(\Phi_t), \text{ for all } v_i \text{ in } G_t\right\},$$

and let $\mathcal{V}$ be the set of vertexes in the input graph that are associated with variables in $\text{MB}_{\mathcal{B}'}(\Phi_t)$, i.e.

$$\mathcal{V} = \left\{v_i : \exists V_i \in \text{MB}_{\mathcal{B}'}(\Phi_t)\right\}. \tag{6}$$

If for all $v_i \in \mathcal{V}$, there exists a subset $\mathcal{M}' \subseteq \mathcal{M} \backslash \mathcal{F}_{\phi,t}(v_i)$ such that

$$\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} F_{v_i}^{(j)} \Big| \mathcal{M}', \ \forall F_{v_i}^{(j)} \in \mathcal{K}_{\Phi_t}(v_i) \backslash \mathcal{F}_{\phi,t}(v_i), \tag{7}$$

where $\mathcal{F}_{\phi,t}(v_i) = \left\{F_{v'}^{(j)} \in \text{MB}_{\mathcal{B}'}(\Phi_t) : v' = v_i, j \in [0..d_{\zeta(v_i)}]\right\}$ and $\mathcal{K}_{\Phi_t}(v_i)$ is defined in Equation (5), then $\mathcal{M}$ is a Markov blanket of Φ_t in $\mathcal{B}^*$.

Due to space limitation, we have to present the proof of this lemma in the Supplementary Materials [1].

3.2.4 Testing Order and Stop Condition for Blanket Growing. Recall the opportunities for reducing number of conditional independence tests discussed earlier, besides variable screening, a superb order for testing the variables and a stop condition can also reduce the search space of the GS algorithm. The surrogate network is specifically designed to make use of these two opportunities. Local structures in $\mathcal{B}'$ allow pruning features that are irrelevant to the prediction by testing the underlying vertex first. We first prove if there exists a feature on a vertex that is not independent from the prediction conditioned on a set Z of other random variables, the corresponding synthetic variable is also not independent to the prediction conditioned on Z; meanwhile, it is also true that if a vertex is associated with a synthetic variable not independent to the prediction conditioned on Z, feature variables on it are also not independent to the prediction conditioned Z.

Lemma 3.4. *In the surrogate network $\mathcal{B}'$, for any synthetic variable V_i, let Z be a set of variables that does not contain any their its grandparent feature variable $F_{v_i}^{(\cdot)}$'s or contribution variables, if there exists some $F_{v_i}^{(j)}$ such that $F_{v_i}^{(j)} \not\perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$, then $V_i \not\perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$.*

Proof of this lemma is presented in the Supplementary Materials [1]. Furthermore, we have the below lemma to show testing a synthetic variable of a vertex can be used to prune features variables that are certainly not in the Markov blanket.

Lemma 3.5. *In the surrogate network $\mathcal{B}'$, for any synthetic variable V_i, let Z be a set of variables that does not contain any of their grandparent feature variable $F_{v_i}^{(\cdot)}$'s or contribution variables. If $V_i \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$, then all its grandparent feature variables and parent contribution variables are independent from the prediction variable conditioned on Z, namely, $F_{v_i}^{(j)} \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$ and $\xi_i^j \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$ for all $j \in [0..d_{\zeta(v_i)}]$.*

Proof of this lemma is also included in the Supplementary Materials [1]. Lemma 3.5 allows one to avoid conducting $O(d_{\zeta(v_i)})$ tests to show that all features on vertex v_i are conditionally independent from the prediction; instead, only one test on V_i is enough.

According to Definition 2.1, the Markov blanket is not unique, because a superset of a Markov blanket is also a Markov blanket of

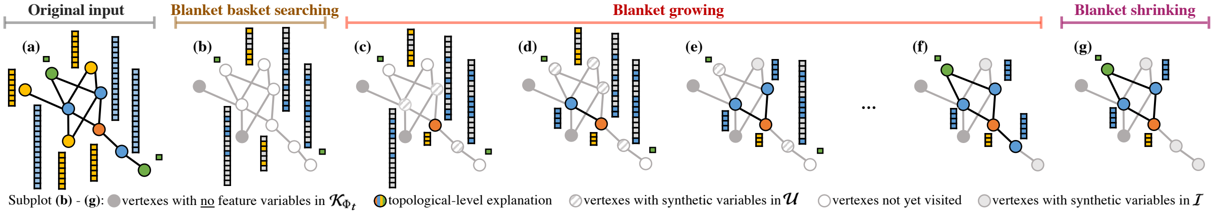

Original input
Blanket basket searching
Blanket growing
Blanket shrinking
(a)
(b)
(c)
(d)
(e)
...
(f)
(g)
Subplot (b) - (g): ● vertexes with no feature variables in $\mathcal{K}_{\Phi_t}$ ● topological-level explanation ▨ vertexes with synthetic variables in $\mathcal{U}$ ○ vertexes not yet visited ● vertexes with synthetic variables in $\mathcal{I}$

Figure 3: Flow of HENCE-X explaining a heterogeneous DGN for node classification in the DBLP dataset. The citation network is a heterogeneous graph with three different types of nodes: *author*, *paper* and *conference*. (a) Original input of the target. (b) Blanket basket searching: the colored features are those in the blanket basket, the others are in gray. (c) The growing phase starts, HENCE-X identifies the target's feature-level explanation (*shorten vector*); conditional independence tests are conducted on synthetic variables in $\mathcal{U}$, which are associated with vertexes (*strip-filled*) in the neighborhood. (d) A 1-hop neighbor, *i.e.* a *paper* is added to $\mathcal{E}_{topo}$ and its feature-level explanation is identified; accordingly, the neighborhood and $\mathcal{U}$ are expanded. (e) Another *paper* is added to $\mathcal{E}_{topo}$, the non-target common author of the two papers (*in light gray*) is tested independent and added to $\mathcal{I}$. (f) the growing phase continues until all neighboring synthetic variables are conditionally independent from the prediction, *i.e.* $|\mathcal{U}| = 0$. (g) The shrinking phase starts, one *unnecessary paper* is removed, a Markov blanket is reached.

the target variable, hence to capture truly important variables, the *necessary variable* is defined as below:

Definition 3.1 (necessary variable). *Consider a joint distribution and a target random variable, a necessary variable is a random variable that is in every Markov blanket of the target variable.*

Specifically in the settings of our problem, we further term vertexes associated with necessary variables V_i in $\mathcal{B}'$ as *necessary vertexes*. To explain DGNs, it is generally believed that the explanation subgraph should be a connected component [21, 54, 57]. The reason is that the disconnected explanation violates the message passing principle of DGNs, as features of disconnected nodes cannot be propagated to the target and thus cannot influence the model's predictions. Based on this philosophy, we assume that *every necessary vertex is connected to the target by other necessary ones*. The meaning of this assumption is that vertexes comprising critical message passing paths are all necessary in determining the prediction variable in the model behavior distribution. In addition, the role of necessary vertexes is to carry the most influential neural information flow to ensure important features can be propagated to the target. Subsequently, a graph traversal strategy can be employed to properly order the variables for conditional independence tests, and an elegant stop condition can be developed.

Testing Order of Variables. As target node t must be included in the explanation, iteratively picking variables for growing a blanket should begin with features of the target, followed by those in the neighborhood. To start with, feature variables in the blanket basket on the target node will be added to the empty blanket for initialization. In the following steps, instead of testing the rest random variables in an arbitrary order, synthetic variables associated with neighbors of the current structure are tested first. This is followed by feature variables on the vertex to be added to the growing blanket in each iteration. As a result, features on nodes that are close to the target will be tested and added to the blanket before those distant from the target. Though it cannot be theoretically proved that such ordering can ensure variables in the Markov blanket are tested foremost due to the DGNs being complex black-box functions, it is generally believed that the closer the nodes are to a target, the more influential they are in the DGN's prediction on the target. In the meantime, connectivity of the explanation structure is certainly well-maintained in this graph expansion strategy. Essentially, testing order is guided by the input graph topology and surrogate network $\mathcal{B}'$ alliteratively. When adding a vertex variable, the graph topology is followed; while adding feature variables, the local structure of $\mathcal{B}'$ is followed.

Stop Condition. Recall that necessary vertexes are all connected to the target, thus the stop condition is to end the growing procedure when there are no variables in the neighborhood that Φ_t is not independent of conditioned on the current blanket. When done this way, no more conditional independence test on other variables will be conducted. Below lemma states such a stop condition guarantees that the untested variables are not in a Markov blanket of Φ_t.

Lemma 3.6. *Given a pretrained DGN ϕ and a target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_\phi(t)$ and every necessary vertex in G_t is connected to the target node through other necessary vertex(es). Let $\mathcal{E}'$ be a set of vertexes, whose induced subgraph of G_t is a connected component that contains the target t. Denote its neighborhood as $\boldsymbol{Ne}(\mathcal{E}') = \bigcup_{v \in \mathcal{E}'} Adj[v] \Big\backslash \mathcal{E}'$, where $Adj[v]$ is the set of all adjacent neighbor(s) of the vertex v; and denote the set of synthetic variables and feature variables induced by $\mathcal{E}'$ as*

$$\mathcal{Z}(\mathcal{E}') = \Big\{V_i : v_i \in \mathcal{E}' \cup \boldsymbol{Ne}(\mathcal{E}')\Big\} \bigcup \Big\{F^{(j)}_{v'} \in \mathcal{K}_{\Phi_t}(v') : v' \in \mathcal{E}' \cup \boldsymbol{Ne}(\mathcal{E}')\Big\}. \tag{8}$$

If for all $V' \in \{V' : v' \in \boldsymbol{Ne}(\mathcal{E}')\}$, there exists a subset $\mathcal{M}' \subseteq \mathcal{Z}(\mathcal{E}')$ such that $\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} V' \Big| \mathcal{M}'$, or $\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} F^{(j)}_{v_i} \Big| \mathcal{M}'$ for all grandparent feature variables of V', then a Markov blanket of Φ_t can be found as a subset of $\mathcal{Z}(\mathcal{E}')$.

Please see the Supplementary Materials [1] for the proof. In conclusion, the strategy is to begin with feature variables on the target node t, then dynamically expand the explanation by adding variables from the neighborhood of the current structure, until there is no such variable in the neighborhood that Φ_t is not conditional independent of.

3.2.5 *Algorithm.*

The pseudo code for HENCE-X is shown in Algorithm 1. The algorithm consists of two phases for building a Markov blanket: the growing phase and the shrinking phase. In the former, following the previously discussed strategy, the proposed explainer starts from the target node, then visits the neighborhood step by step. In each iteration, HENCE-X picks the most promising vertex and inserts it into the topological-level explanation, then HENCE-X continues to find feature-level explanation for the vertex.

Algorithm 1: HENCE-X

```
Input: the pretrained DGN model $\phi$, a target node $t$ and its computational graph $G_t$;
Output: a set of selected variables;
Initialization: $\mathcal{E}_{topo} \leftarrow \varnothing$, $\mathcal{U} = \{T\}$, $\mathcal{I} \leftarrow \varnothing$, $\mathcal{M} \leftarrow \varnothing$;
Generate $\mathcal{D}_{\mathcal{P}_{\phi}(t)}$;
$\mathcal{K}_{\Phi_t} \leftarrow Equation$ (3);
Build $\mathcal{B}'$ based on $\mathcal{K}_{\Phi_t}$;
/* Growing phase, add variables to the blanket.                */
while $|\mathcal{U}| \neq 0$ and $\exists V \in \mathcal{U}$ s.t. $\Phi_t \not\perp\!\!\!\perp_{\mathcal{D}_{\mathcal{P}_{\phi}(t)}} V \Big| \mathcal{M}$ do
    $\hat{V} = \operatorname*{argmax}_{V \in \mathcal{U}}$ dependency of $\Phi_t$ on $V$ conditioned on $\mathcal{M}$;
    $\mathcal{I} \leftarrow \mathcal{I} \bigcup \Big\{ V \in \mathcal{U} :$ s.t. $\Phi_t \perp\!\!\!\perp_{\mathcal{D}_{\mathcal{P}_{\phi}(t)}} V \Big| \mathcal{M} \Big\}$;
    Remove $\hat{V}$ from $\mathcal{U}$;
    $\hat{\mathcal{M}} \leftarrow \Big\{ F^{(j)}_{\hat{v}} \in \mathcal{K}_{\Phi_t} : \Phi_t \not\perp\!\!\!\perp_{\mathcal{D}_{\mathcal{P}_{\phi}(t)}} F^{(j)}_{\hat{v}} \Big| \mathcal{M} \Big\}$;
    if $|\hat{\mathcal{M}}| = 0$, then
        $\mathcal{I} \leftarrow \mathcal{I} \bigcup \{\hat{V}\}$
    else
        $\mathcal{E}_{topo} \leftarrow \mathcal{E}_{topo} \cup \{\hat{v}\}$;
        $\mathcal{E}^{\hat{v}}_{feat} \leftarrow \Big\{ f^{(j)}_{\hat{v}} : F^{(j)}_{\hat{v}} \in \hat{\mathcal{M}} \Big\}$;
        $\mathcal{M} \leftarrow \mathcal{M} \bigcup \hat{\mathcal{M}}$;
        $\mathcal{U} \leftarrow \mathcal{U} \bigcup \Big\{ V' : v' \in Adj[\hat{v}]$ and $v' \notin \mathcal{E}_{topo} \Big\}$;
    $\mathcal{U} \leftarrow \mathcal{U} \backslash \mathcal{I}$;
/* Shrinking phase, remove unnecessary variable(s).     */
for $v \in \mathcal{E}_{topo}$ do
    if $\Phi_t \perp\!\!\!\perp_{\mathcal{D}_{\mathcal{P}_{\phi}(t)}} V \Big| \Big\{ F^{(j)}_{v'} : f^{(j)}_{v'} \in \bigcup_{v'' \in \mathcal{E}_{topo} \backslash \{v\}} \mathcal{E}^{v''}_{feat} \Big\}$ then
        $\mathcal{E}_{topo} \leftarrow \mathcal{E}_{topo} \backslash \{v\}$;
return $\mathcal{E}_{topo}, \mathcal{E}^{v}_{feat}$'s for all $v \in \mathcal{E}_{topo}$.
```

However, in the growing phase there may be some variables added by the algorithm, but they are actually not necessary for the blanket. These variables are independent from the prediction conditioned on a grown blanket in a later iteration [25]. Thus we also introduce the shrinking phase to eliminate such effect while preserving the completeness of the Markov blanket, which will be proved in the next subsection.

An example of how HENCE-X explains a heterogeneous DGN for node classification in DBLP dataset is illustrated in Figure 3. The citation network is a heterogeneous graph with three different types of nodes: ***author***, ***paper*** and ***conference***. In *Subplot* **(a)** shows the original input of the **target**. The blanket basket searching is shown in *Step* **(b)**, features that remain colored are included in the blanket basket, the others are in gray. The growing phase is illustrated from *Step* **(c)** to *Step* **(f)**. In *Step* **(c)**, HENCE-X starts from the **target** and identifies its feature-level explanation $\hat{\mathcal{M}}$, then conducts conditional independence tests on synthetic variables in $\mathcal{U}$, which are associated with vertexes (*striped* in the figure) in the neighborhood. In *Step* **(d)**, the explainer chooses a 1-hop neighbor, *i.e.*, a ***paper*** to add to $\mathcal{E}_{topo}$ and finds its feature-level explanation; accordingly, the neighborhood and $\mathcal{U}$ are expanded. In *Step* **(e)**, another ***paper*** is added to $\mathcal{E}_{topo}$, while the non-target common author (in **light gray**) of the two papers is tested conditionally independent and added to $\mathcal{I}$. The growing phase continues until all neighboring synthetic variables are conditionally independent from the prediction, *i.e.*, $|\mathcal{U}| = 0$, as shown in *Step* **(f)**. The shrinking phase is then carried out in *Step* **(g)**, where one unnecessary ***paper*** is removed and HENCE-X reaches a Markov blanket.

3.2.6 Theoretical Guarantee of the Algorithm. We claim that that Algorithm 1 can produce a Markov blanket of the prediction variable in the original perfect map $\mathcal{B}^*$ of the model behavior distribution, such that the output explanation includes all statistical information that the model's decision is dependent on. Formally, the below theorem provides the desired guarantee.

Theorem 3.7. *Given a pretrained DGN ϕ and a target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_{\phi}(t)$ and every necessary vertex in G_t is connected to the target node through other necessary vertex(es). Running Algorithm 1 to solve the heterogeneity-agnostic multi-level explanation generation problem defined in Definition 2.3 outputs the topological-level explanation $\mathcal{E}_{topo}$ and the feature-level explanation $\mathcal{E}^{v}_{feat}$'s for each vertex v in $\mathcal{E}_{topo}$. Let $\mathcal{M}^* = \Big\{ F^{(j)}_{v_i} : f^{(j)}_{v_i} \in \bigcup_{v' \in \mathcal{E}_{topo}} \mathcal{E}^{v'}_{feat} \Big\}$, then $\mathcal{M}^*$ is a Markov blanket of Φ_t in $\mathcal{B}^*$.*

Please refer to the Supplementary Materials [1] for the proof.

3.3 Factual and Counterfactual Explanation

By modeling the DGN's behavior using a Bayesian network, one can easily analyze the two different cases where the prediction confidence is maximized or minimized. When $\Phi_t = max$, the values of feature variables correspond to a *factual explanation* [23, 54], which can maximize the model's belief in its decision by presenting *only* the critical parts. When $\Phi_t = min$, the realization of features can be considered a *counterfactual explanation* [21, 22], which can minimize the model's confidence by removing important information. The two kinds of explanation can be found using the parameters in the Bayesian network. The surrogate network $\mathcal{B}'$ allows us to prove Algorithm 1 can find all the direct parents of the prediction variable in $\mathcal{B}'$. This fact is proved by the below theorem.

Theorem 3.8. *Given a pretrained DGN ϕ and a target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_{\phi}(t)$ and every necessary vertex in G_t is connected to the target node through other necessary vertex(es). Running Algorithm 1 to solve the heterogeneity-agnostic multi-level explanation generation problem defined in Definition 2.3 outputs the topological-level explanation $\mathcal{E}_{topo}$. Let $\mathcal{R} = \{V_i : \exists v_i \in \mathcal{E}_{topo}\}$, then $\mathcal{R}$ is the set of all the direct parents of Φ_t in $\mathcal{B}'$, i.e. $\mathcal{R} = \boldsymbol{Pa}(\Phi_t)$.*

Proof of this theorem can be found in the Supplementary Materials [1]. According to Theorem 3.8, the maximum likelihood estimation (MLE) can be used to learn parameters of the parent synthetic variables of the prediction variable in $\mathcal{B}'$. Recall that as long as the one-to-one mapping from the parent contribution variables to the realization of a synthetic variable is well-maintained, finding the maximal parameters of synthetic variables in the Bayesian network also tells us the underlying realization of feature variables. Mathematically, the parameters of different realization of Φ_t's parents can be estimated by $\theta = Pr(\Phi_t | \boldsymbol{Pa}(\Phi_t))$ using MLE. Given the Markov blanket, the realization of variables in $\mathcal{R}$ with the highest parameters for the prediction being maximized or minimized are considered to be a factual or a counterfactual explanation. Formally,

$$^{F}\mathcal{E} = categorize^{-1}(\operatorname*{argmax}_{\mathcal{R}} Pr(\Phi_t = max | \boldsymbol{Pa}(\Phi_t))),$$

$$^{CF}\mathcal{E} = categorize^{-1}(\operatorname*{argmax}_{\mathcal{R}} Pr(\Phi_t = min | \boldsymbol{Pa}(\Phi_t))),$$

4 EXPERIMENTAL EVALUATION

We evaluate our proposed HENCE-X on three real-world datasets from different domains against SOTA baselines, including both heterogeneous and homogeneous graphs. In the following, we first introduce the quantitative metrics used in the evaluation, then report the details on datasets used in the experiments and different DGN models to be explained.

4.1 Evaluation Metrics

To measure the effectiveness of factual and counterfactual explanations respectively, we introduce two corresponding metrics. Let $\mathcal{E}_{topo}$ be the topological-level explanation and $\mathcal{E}_{feat} = \bigcup_{v \in \mathcal{E}_{topo}} \mathcal{E}^{v}_{feat}$ be the corresponding feature-level explanation. The *factual effect* is defined to evaluate the factual explanation and calculated as below:

$$\boldsymbol{F_{effect}}(\phi, t, \mathcal{E}_{topo}, \mathcal{E}_{feat}) = \phi\big(G_t, \mathcal{X}^{+}_{\mathcal{E}_{feat}}\big),$$

where $\mathcal{X}^{+}_{\mathcal{E}_{feat}}$ stands for perturbed node features that only keep those in $\mathcal{E}_{feat}$ while the others are occluded. $\boldsymbol{F_{effect}}$ is designed to evaluate the explanation by answering "*how confidence is the DGN when given only the critical part of the information?*". Notice that here we do not modify the adjacency matrix in the input, because structural change commonly results in a disastrous DGN confidence drop, thus we follow Yuan, *et al.* [57] to use zero-padding for occluding important nodes, *i.e.* set features in the explanation to be zero to eliminate their contribution. Meanwhile, the *counterfactual effect* is designed to evaluate the counterfactual explanation as below:

$$\boldsymbol{CF_{effect}}(\phi, t, \mathcal{E}_{topo}, \mathcal{E}_{feat}) = \phi(G_t, \mathcal{X}) - \phi(G_t, \mathcal{X}^{-}_{\mathcal{E}_{feat}})$$

where $\mathcal{X}^{-}_{\mathcal{E}_{feat}}$ represents node features obtained from zero-padding out features in $\mathcal{E}_{feat}$. $\boldsymbol{CF_{effect}}$ is introduced to answer "*how much does the DGN's confidence drop without the critical part of the information?*". For both metrics, the higher the better.

Besides, a supreme explanation should also be concise and precise. Consider taking almost the entire original input as explanation, $\boldsymbol{F_{effect}}$ and $\boldsymbol{CF_{effect}}$ are both expected to be high as the explanation contains almost all the information, yet such an explanation is meaningless. Thus one needs to measure the size of explanations as well. We define *density of topological-level explanation* as below:

$$\boldsymbol{T_{Dens.}}(t, \mathcal{E}_{topo}) = \frac{\|\mathcal{E}_{topo}\|}{\|\mathcal{V}\|}.$$

Similarly, *density of feature-level explanation* is evaluated as below:

$$\boldsymbol{F_{Dens.}}(t, \mathcal{E}_{topo}, \mathcal{E}_{feat}) = \frac{\sum_{v \in \mathcal{E}_{topo}} \|\mathcal{E}^{v}_{feat}\| / d_{\zeta(v)}}{\|\mathcal{E}_{topo}\|},$$

which is the average portion of features selected as explanation among all nodes in the topological-level explanation. Furthermore, we measure the overall density of the multi-level explanation by calculating the weighted sum using the below metric.

$$\boldsymbol{O_{Dens.}}(t, \mathcal{E}_{topo}, \mathcal{E}_{feat}) = \frac{\sum_{v \in \mathcal{E}_{topo}} \|\mathcal{E}^{v}_{feat}\| / d_{\zeta(v)}}{\|\mathcal{V}\|}.$$

For density metrics, the lower the better.

Dataset-DGN	Node	Edge	Labels
DBLP-HGT *Node classification* *Train. acc.* 1.0 *Test. acc.* 0.7872	# author (A): 4,057 # paper (P): 14,328 # venue (V): 20 *avg.* in *CG*: 9.31	# A-P: 19,645 # P-V: 14,328	Database Data Mining Artificial Intelligence Information Retrieval
IMDB-HAN *Node classification* *Train. acc.* 0.990 *Test. acc.* 0.4902	# movie (M): 4,278 # director (D): 2,081 # actor (A): 5,257 *avg.* in *CG*: 22.81	# M-D: 4,278 # M-A: 12,828	Action Comedy Drama
MUTAG-GCN *Graph classification* *Train. 0.8412.* *Test. acc.* 0.9444	*avg.* 17.93	*avg.* 19.79	Mutagenic (Non-mutagenic*)

Table 1: Statistics of datasets and details of DGN models. *CG* is short for computational graph of each instance in the dataset.

4.2 Datasets and DGNs to Be Explained

Three widely-used datasets are used in the experiment: DBLP and IMDB datasets are adopted for the node classification and the MUTAG dataset is used for the graph classification. We train three different DGNs on the three datasets. Statistics of the datasets and details of DGN models are summarized in Table 1.

4.2.1 *Datasets.*

4.2.1 Datasets. Below we introduce the details of the datasets.

DBLP dataset (*heterogeneous graph*). DBLP[4] is a bibliography website in the area of computer science. We employ a subset of the DBLP dataset used by related heterogeneous DGNs [8, 49]. There are 4,057 authors, 14,328 papers and 20 conferences (venues) in the citation network. The authors are labeled by one of the four areas: Database (DB), Data Mining (DM), Artificial Intelligence (AI), and Information Retrieval (IR). Bag-of-words representation is used for author node and paper node to encode their research keywords and words in the title, respectively. For conference nodes, a uniform 1-dimension feature with value 1 is used as their initial feature.

IMDB dataset (*heterogeneous graph*). IMDB[5] is a website about movies and television shows, which contains information about cast, production crew, plot summaries, and the like. We adopt the IMDB dataset used by Wang *et al.* [49] and Fu *et al.* [8] for heterogeneous DGNs. The dataset includes 4,278 movies and 2,081 directors in addition to 5,257 actors. Movies are classified by their genres: Action, Comedy, or Drama. Movie nodes use bag-of-words representation of their plot keywords as the initial features.

MUTAG dataset (*homogeneous graph*). The dataset contains the molecular structure of chemical compounds classified by their mutagenic effect on a bacterium [6]. In each piece of graph, nodes represent different atoms and edges stand for chemical bonds. There are a total of 7 chemical elements: Carbon, Nitrogen, Oxygen, Fluorine, Iodine, Chlorine and Bromine. One-hot encoding of atoms' chemical elements is used as node features.

4.2.2 DGNs to be explained. We train two heterogeneous DGNs and one DGN to be explained. Below we detail their structure.

HGT [12]. Heterogeneous graph transformer (HGT) is a transformer-based DGN equipped with subgraph sampling, which has demonstrated outstanding performances on web-scale graphs. The model is based on triplet-level attention mechanism. We build a model with one linear transformation layer followed by two HGT layers and one more linear layer for node classification task. The number

[4] https://dblp.org

[5] https://www.imdb.com

	DBLP						IMDB						MUTAG					
Method	F_{effect}	CF_{effect}	$O_{Dens.}$	$T_{Dens.}$	$F_{Dens.}$	$Time_{avg}$	F_{effect}	CF_{effect}	$O_{Dens.}$	$T_{Dens.}$	$F_{Dens.}$	$Time_{avg}$	F_{effect}	CF_{effect}	$O_{Dens.}$	$T_{Dens.}$	$F_{Dens.}$	$Time_{avg}$
^{F}HENCE-X$_{k=10}$	0.9888	-	0.1829	0.2963	0.6060	18.72s	0.8501	-	0.0763	0.1343	0.5728	18.89s	0.6676	-	0.0629	0.4406	0.1429	20.90s
^{F}HENCE-X$_{k=15}$	**0.9909**	-	**0.1452**	0.2570	0.5540	22.62s	0.8503	-	0.0770	0.1344	**0.5732**	24.59s	0.6621	-	0.0605	0.4236	0.1429	18.99s
^{F}HENCE-X$_{k=20}$	0.9881	-	0.1778	0.3017	**0.5735**	28.15s	0.8506	-	0.0771	**0.1341**	0.5771	33.66s	0.6608	-	0.0622	0.4352	0.1429	18.67s
^{F}HENCE-X$_{k=25}$	0.9886	-	0.1771	0.3029	0.5744	36.05s	**0.8507**	-	0.0770	0.1342	0.5782	43.11s	**0.6700**	-	0.0628	0.4393	0.1429	20.83s
^{F}HENCE-X$_{k=30}$	0.9894	-	0.1803	0.3055	0.5777	42.29s	**0.8507**	-	0.0770	0.1344	0.5764	51.69s	0.6659	-	0.0655	0.4588	0.1429	15.34s
^{F}HENCE-X$_{avg}$	*0.9892*	-	*0.1727*	*0.2927*	*0.5771*	*29.57s*	*0.8504*	-	*0.0769*	*0.1343*	*0.5749*	*34.39s*	*0.6653*	-	*0.0628*	*0.4395*	*0.1429*	*18.95s*
^{CF}HENCE-X$_{k=10}$	-	0.8757	0.1725	0.2887	0.5785	18.72s	-	0.5827	0.0869	0.1347	0.6350	18.89s	-	0.1607	0.0337	0.2360	0.1429	20.90s
^{CF}HENCE-X$_{k=15}$	-	**0.8853**	0.1544	**0.2527**	0.5969	22.62s	-	0.5835	0.0870	0.1349	0.6350	24.59s	-	**0.1647**	0.0328	**0.2297**	0.1429	18.99s
^{CF}HENCE-X$_{k=20}$	-	0.8776	0.1805	0.2946	0.5960	28.15s	-	0.5836	0.0873	0.1350	0.6362	33.66s	-	0.1661	0.0339	0.2373	0.1429	18.67s
^{CF}HENCE-X$_{k=25}$	-	0.8807	0.1808	0.2958	0.5973	36.05s	-	**0.5839**	0.0864	0.1352	0.6339	43.11s	-	0.1601	0.0334	0.2335	0.1429	20.83s
^{CF}HENCE-X$_{k=30}$	-	0.8794	0.1815	0.2982	0.5952	42.29s	-	0.5838	0.0873	0.1352	0.6345	51.69s	-	0.1589	**0.0331**	0.2319	0.1429	15.34s
^{CF}HENCE-X$_{avg}$	-	*0.8797*	*0.1740*	*0.2860*	*0.5928*	*29.57s*	-	*0.5835*	*0.0870*	*0.1350*	*0.6349*	*34.39s*	-	*0.1621*	*0.0334*	*0.2337*	*0.1429*	*18.95s*
GNNExplainer	0.9644	0.6646	0.3977	0.4889	0.6941	1.71s	0.8369	0.5664	0.1974	0.3512	0.5991	1.06s	0.0771	0.0233	0.1465	0.3417	0.4286	0.56s
CF^{2}	0.9008	0.7049	0.4437	0.5803	0.6010	1.84s	0.8395	0.4919	0.2325	0.3668	0.5836	1.13s	0.0631	0.0304	0.1196	0.2792	0.4286	0.51s
GCN-LRP	-	-	-	-	-	-	-	-	-	-	-	-	0.1086	0.0418	0.1439	0.3358	0.4286	1.27s
PGM-Explainer	0.7836	0.6955	-	0.5431	-	21.64s	0.4202	0.3425	-	0.3896	-	23.11s	0.1674	0.1333	-	0.3529	-	14.20s
PGExplainer	0.8495	0.7198	-	0.5763	-	**0.02s**	0.7896	0.5219	-	0.2999	-	**0.007s**	0.5596	0.1013	-	0.6887	-	**0.001s**
SubgraphX	0.7867	0.1679	-	0.6602	-	1317s	0.7727	0.0001	-	0.8849	-	525.45s	0.4321	0.1088	-	0.3305	-	96.01s
Gem	0.8601	0.6299	-	0.5603	-	2.34s	0.7871	0.5073	-	0.6380	-	1.73s	0.6310	0.1304	-	0.2792	-	0.49s
RG-Explainer	0.8917	0.7369	-	0.5893	-	17.29s	0.8004	0.5359	-	0.5830	-	12.94s	0.5725	0.1274	-	0.4118	-	7.80s

Table 2: Quantitative evaluation results. HENCE-X outperforms the competitors regarding effectiveness. Though HENCE-X is not the most efficient explainer, the running time is reasonable. Please see Section 4.4 for more discussion.

of attention heads is 2 and the hidden dimension is set to 64. The semi-supervised model achieves a training accuracy that ensures it has learned the knowledge in the training set (see Table 1).

HAN [49]. Heterogeneous graph attention network (HAN) is a meta-path-based DGN that utilizes a hierarchical attention mechanism: semantic- and node- level. Given a meta-path, the node-level attention is calculated by a Graph Attention Network [46] from neighbors connected by the meta-path; then, the semantic-level attention is computed as a weighted average of the node-level attention from all meta-paths. We build a HAN model with one linear layer as the final predictor for the node classification task. The number of attention heads and the dimension of hidden channel are set to 8 and 128, accordingly. Similar to HGT, we train HAN to a sufficient training accuracy (see Table 1).

GCN [14]. Graph Convolution Network extends convolution techniques in image data to the graph setting, which is a stacking-style homogeneous DGN. The model used in the experiment aims for the graph classification task, which consists of 3 GCN layers followed by a global pooling layer and 2 fully connected layers as the final predictor. ReLU is applied on the output of the first two GCN layers, while the dropout is utilized between every two GCN layers and the two fully connected layers, with the rates all set to 0.5.

4.3 Competitors and Algorithm Settings

Multi-level Competitors. We compared HENCE-X to three existing multi-level explainers.

- GNNExplainer [54] learns an edge mask and a node feature mask that maximizes Mutual information between model's predictions on the original input and predictions of the masked input.
- CF^2 [45] uses the same masking strategy as GNNExplainer's with an objective that considers factual and counterfactual reasoning.
- GCN-LRP [11] is an explainer specifically developed for GCN [14] only based on layer-wise relevance propagation (LRP) [2].

The first two methods are designed for homogeneous graphs only. Thus, for the MUTAG dataset, we evaluate them using the original method. For the other two heterogeneous graphs, we implement a modified version of them to allow several masks for different types of nodes, such that each feature space corresponds to one specific mask, whose shapes are different due to mismatched feature dimensions. However, GCN-LRP is designed for GCN only, hence we test the technique on the MUTAG dataset but not the other two.

Topological-level Competitors. We further compare our method to five topological-level methods.

- PGM-Explainer [47] is a Bayesian-network-based method that treats *vertexes* in the input graph as random variables.
- PGExplainer [23] is an inductive explainer, which can be directly used on new instances after training on a group of data.
- SubgraphX [57] adopts the Shapley value [17] as the importance measurement and explores subgraphs to find explanations.
- Gem [21] trains an encoder-decoder network to generate adjacency matrices as explanations based on supervised learning.
- RG-Explainer [42] utilizes a reinforcement learning framework with three learned modules, *i.e.*, seed locator, graph generator and stopping criteria, to generate explanations.

Variants of HENCE-X. We conduct ablation studies for the *variable screening* (VS) module in Section 3.2.1 and *testing order and stop/condition* (TS) module in Section 3.2.4. Note that blanket shrinking can not be skipped because Theorem 3.8 requires the output synthetic variables to be the direct parents of the prediction variable. Furthermore, we evaluate HENCE-X using the entire sample space as the sample set, *i.e.,* the set of all possible outcomes from perturbing the input.

Dataset	#FeatRV	k=10	k=15	k=20	k=25	k=30
DBLP	80.03	1292.62	1593.70	1932.53	2291.38	2663.70
IMDB	146.89	1642.88	2342.87	3065.83	3798.64	4537.04
MUTAG	19.94	1000.00	1000.00	1000.00	1000.00	1000.00

Table 3: Average number of feature variables in the model behavior distribution and sample size *v.s.* different k values among all instances in each dataset.

- HENCE-X$_{\text{VS}^-}$ is a variant of HENCE-X whose blanket basket searching step (VS module) is removed
- HENCE-X$_{\text{TS}^-}$ is a variant of HENCE-X whose TS module is removed. Note that if removing the TS module, HENCE-X no longer requires the output explanation to be a connected components.
- HENCE-X* is the original HENCE-X that exhaustively enumerate all possible perturbing situations to generate samples.

Due to these variants being very time-consuming, we employ a subset of the DBLP dataset, named dblp, consisting of instances with small numbers of random variables, such that explaining each single instance can be finished in *one hour*. The same scheme is used to construct a subset of the MUTAG dataset, named mutag. Statistics of the two small datasets are presented in Table 4.

4.3.1 Setting for explainers. Grid Search is applied for all methods in the experiments. Specifically, we start from the default parameter settings of each competitor in the corresponding task (node or graph classification), then search for better parameters using grid size as the one tenth of one in the last nonzero digit in the default value. For example, if the start point is 0.5, the grid size will be 0.01.For HENCE-X, there are two parameters to search, *i.e.*, the probability of perturbing features for sample generation and σ in Equation (2), the starting places for them are 0.5 and 0.01, respectively. G-test [26] is employed in Algorithm 1 to conduct conditional independence tests for all datasets, the statistical significance is set to a conventional criteria 0.05. The sample size is the critical parameter for HENCE-X that affects the performance of independence-test-based methods. We follow the rule of thumb to set the sample size for multivariate distribution: the number of samples should be several times as large as the number of variables, preferably ten times or more [38, 41]. Let k be the multiple in determining the sample number, we test HENCE-X using k from 10 to 30 using a step of 5, *i.e.* $k \in \{10, 15, 20, 25, 30\}$ for comparison. Furthermore, if the sample size is smaller than 1000, it will be increased to 1000 as suggested by McDonald (2014) [27]. The sample size and the average number of feature variables of each instance are reported in Table 3.

4.4 Experimental Result

4.4.1 Quantitative evaluation. The quantitative evaluation results are reported in Table 2. Regarding effectiveness, HENCE-X outperforms all of its competitors. It is noted that on the MUTAG dataset, the factual results of HENCE-X have higher topological densities than topological-level explainers; yet, these explainer takes all features on the nodes as part of the explanation, hence if weight T_{Dens} by 1 as their overall density, HENCE-X is less dense. As for running time, PGExplainer is the one with shortest testing time. However, as an inductive explainer, it requires 38480.20s and 4376.92s training time on the DBLP and IMDB datasets, respectively (see Table 6 in Supplementary Material [1]). As explaining DGNs at instance level is not efficiency-demanding because it serves individual instances, and human understanding of the output is involved, we believe the running time for HENCE-X is reasonable. Evaluation on variants of HENCE-X is reported in Table 4. HENCE-X$_{k=10}$ outperforms HENCE-X* in all metrics by limited margins, while taking a much shorter time. The two settings lead to different results, because the sample size affects the sampled DGN score range, which will further affect the realization of the prediction variables using Equation (2). According to the experimental results, we believe using $k \cdot N_{\text{RV}}$ with k no less than 10 for HENCE-X can achieve premier outputs. Nonetheless, we suggest users to use a $k \geq 15$. For the ablation study on variable screening (blanket basket searching) in both the dblp and mutag datasets, HENCE-X outperforms the other variants regarding $\boldsymbol{F_{effect}}$, $\boldsymbol{CF_{effect}}$ and running time. This is because, without the two modules, more feature variables can be added to the Markov blanket in the growing phase yet they are not as important, while the shrinking phase only removes variables at the vertex level, those feature variables then remain in the output blanket; while the two modules both help prune variables from independence test. It is also obvious that HENCE-X is slower without the pruning module. The advantage is not very prominent because instances in the two datasets are rather small in terms of number of random variables, thus we measure the average ratio of pruned feature random variables among all instances in each dataset. The results are shown in Table 5. The basket searching module prunes at least 48% of feature variables on the three datasets, the ratio is higher than 89% on the IMDB dataset.

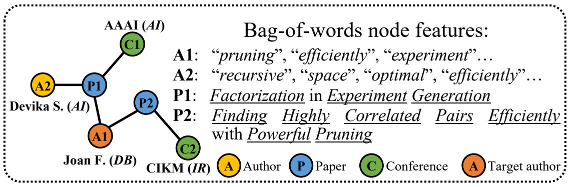
	dblp (*avg.* #FeatRV: 14.5, *avg.* #Nodes in *CG*: 3.875)					
Variant	F_{effect}	CF_{effect}	$O_{Dens.}$	$T_{Dens.}$	$F_{Dens.}$	$Time_{avg}$
$^{\text{F}}$HENCE-X$_{\text{k=10}}$	0.9977	-	0.2838	0.4646	0.6333	9.37s
$^{\text{F}}$HENCE-X*	0.9971	-	0.5273	0.7791	0.6798	1158s
$^{\text{F}}$HENCE-X$_{\text{VS}^-}$	0.9976	-	0.4095	0.6417	0.6507	20.49s
$^{\text{F}}$HENCE-X$_{\text{TS}^-}$	0.9967	-	0.3123	0.5375	0.5708	12.89s
$^{\text{CF}}$HENCE-X$_{\text{k=10}}$	-	0.9373	0.4433	0.6762	0.6817	9.37s
$^{\text{CF}}$HENCE-X*	-	0.9176	0.4502	0.6920	0.6443	1158s
$^{\text{CF}}$HENCE-X$_{\text{VS}^-}$	-	0.8427	0.3501	0.5688	0.6087	20.49s
$^{\text{CF}}$HENCE-X$_{\text{TS}^-}$	-	0.7887	0.3137	0.5062	0.5917	12.89s
	mutag (*avg.* #FeatRV: 15.5, *avg.* # Nodes in *CG*: 14.5)					
Variant	F_{effect}	CF_{effect}	$O_{Dens.}$	$T_{Dens.}$	$F_{Dens.}$	$Time_{avg}$
$^{\text{F}}$HENCE-X$_{\text{k=10}}$	0.8366	-	0.0676	0.4735	0.1428	6.23s
$^{\text{F}}$HENCE-X*	0.8308	-	0.1020	0.7139	0.1429	1839s
$^{\text{F}}$HENCE-X$_{\text{VS}^-}$	0.8350	-	0.0412	0.2885	0.1429	14.67s
$^{\text{F}}$HENCE-X$_{\text{TS}^-}$	0.8350	-	0.0412	0.2882	0.1431	7.73s
$^{\text{CF}}$HENCE-X$_{\text{k=10}}$	-	0.0722	0.0288	0.2019	0.1428	6.23s
$^{\text{CF}}$HENCE-X*	-	0.0717	0.0333	0.2332	0.1429	1839s
$^{\text{CF}}$HENCE-X$_{\text{VS}^-}$	-	0.0363	0.0189	0.1322	0.1429	14.67s
$^{\text{CF}}$HENCE-X$_{\text{TS}^-}$	-	0.0417	0.1093	0.5162	0. 0235	7.73s

Table 4: Results of empirical study on variants of HENCE-X. Meanings of subscript are the following - $k = 10$: original HENCE-X using $k = 10$, VS$^-$: HENCE-X without the *basket searching* module, TS$^-$: HENCE-X without *random variable test ordering and stop condition*; the latter two are ablation studies on HENCE-X. While HENCE-X* denotes the method using the entire population as the sample set.

	DBLP	IMDB	MUTAG
***avg.* ratio of pruned FeatRV**	0.6235	0.8917	0.4863

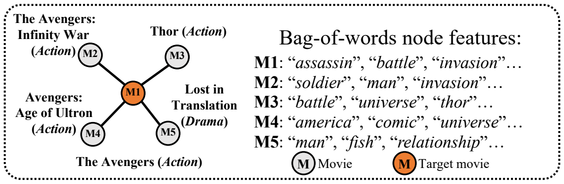
Table 5: Average ratio of pruned feature random variables among all instances in each dataset.

Discussion on HENCE-X in different tasks. HENCE-X is shown to be effective in different tasks. Note that partial explanations may


AAAI (*AI*)
C1
A2
P1
Devika S. (*AI*)
P2
A1
C2
Joan F. (*DB*)
CIKM (*IR*)
Bag-of-words node features:
A1: "*pruning*", "*efficiently*", "*experiment*"…
A2: "*recursive*", "*space*", "*optimal*", "*efficiently*"…
P1: *Factorization* in *Experiment* *Generation*
P2: *Finding* *Highly* *Correlated* *Pairs* *Efficiently* with *Powerful* *Pruning*
A Author
P Paper
C Conference
A Target author

- □ ^{F}HENCE-X: **A1** - "generation", "experiment", "efficiently", "pruning"
- □ ^{CF}HENCE-X: **A1** - "generation", "experiment", "efficiently", "pruning"
- □ GNNExplainer: **A1** - "finding", "generation", "experiment", "efficiently", "pruning", **P2** - "efficiently", "finding", "pair", "pruning", **P1** - "experiment", "factorization", "generation", **C1** – AAAI
- □ CF^2: **A1** - "finding", "generation", "experiment", "efficiently", "pruning", **P2** - "efficiently", "finding", "pair", "pruning", **P1** - "experiment", "factorization", "generation", **C1** – AAAI
- ❖ PGM-Explainer: **A1**, **P1**, **C2**
- ❖ PGExplainer: **A1**, **P1**, **P2**, **C2**
- ❖ SubgraphX: **A1**, **P1**, **P2**, **C1**
- ❖ Gem: **A1**, **P2**, **C2**
- ❖ RG-Explainer: **A1**, **P2**, **C1**

Figure 4: Qualitative result on the DBLP dataset. Box-shape (*w.r.t.* diamond-shape) bullet represents multi-level (*w.r.t.* topological-level) explainers.


The Avengers: Infinity War (*Action*)
M2
Thor (*Action*)
M3
M1
Avengers: Age of Ultron (*Action*)
M4
Lost in Translation (*Drama*)
M5
The Avengers (*Action*)
Bag-of-words node features:
M1: "*assassin*", "*battle*", "*invasion*"…
M2: "*soldier*", "*man*", "*invasion*"…
M3: "*battle*", "*universe*", "*thor*"…
M4: "*america*", "*comic*", "*universe*"…
M5: "*man*", "*fish*", "*relationship*"…
M Movie
M Target movie

- □ ^{F}HENCE-X: **M1** - "assassin", "battle", "invasion", "iron", "man", "soldier"
- □ ^{CF}HENCE-X: **M1** - "assassin", "battle", "invasion", "iron", "soldier"
- □ GNNExplainer: **M1** - "alien", "assassin", "battle", "invasion", "man", "soldier", **M2** - "alien", "assassin", "battle", "invasion", "man", "soldier"
- □ CF^2: **M1**-"alien", "assassin", "battle", "invasion", "man", "soldier", **M3** - "battle", **M4** - "comic"
- ❖ PGM-Explainer: **M1**, **M2**
- ❖ PGExplainer: **M1**, **M2**
- ❖ SubgraphX: **M1**
- ❖ Gem: **M1**, **M2**, **M5**
- ❖ RG-Explainer: **M1**, **M2**

Figure 5: Qualitative result on the IMDB dataset. Box-shape (*w.r.t.* diamond-shape) bullet represents multi-level (*w.r.t.* topological-level) explainers.

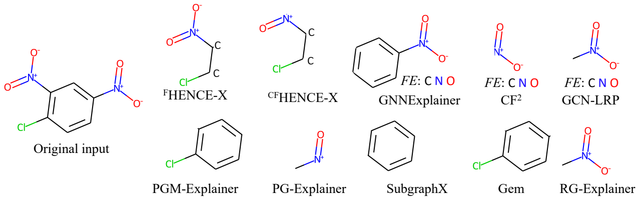


Figure 6: Qualitative results on the MUTAG dataset. *FE* is short for feature-level explanation. Note that *FE* of HENCE-X is node-specific.

be acceptable in the graph classification task because the global pooling layer will aggregate neural messages over the entire graph. Depending on the application, one may drop the TS (random variable test ordering and the stop condition) module in HENCE-X to allow for the disconnected output. Nonetheless, the ablation study in Table 4 shows that connectivity leads to better results.

4.4.2 Qualitative evaluation. Visualization of results on the DBLP, IMDB and MUTAG datasets are shown in Figures 4, 5 and 6, respectively. On the DBLP dataset, we use the same example as in Section 1. HENCE-X ignores the misleading structural information and only focuses on features that are critical, while other explainers fail to do so. HENCE-X also gives precise and concise explanations for instances in IMDB. On the MUTAG dataset, HENCE-X finds the distinct substructure of the input: the Chlorine atom and the nitrogen dioxide structure; moreover, it identities the correct atom types as the node-specific feature explanations, while the other multi-level explainers only find frequent atoms (Carbon, Nitrogen and Oxygen) as a uniform feature explanation for all nodes in the output structure; they also miss the distinct Chlorine atom. While the topological-level explainers fail to capture the two distinct chemical structures simultaneously.

5 RELATED WORK

Deep Graph Networks incorporate graph topology and node features by aggregating neural messages from the neighbors. Renowned homogeneous GNNs include graph convolution networks (GCNs) [14], graph attention networks (GATs) [46], and graph isomorphism networks (GINs) [53]. HGNNs can be grouped into meta-path-based methods and stacking-fashion models according to their way of entangling edge and node types into the network. The former relies on meta-paths to transform the input graph into a heterogeneity-aware computational graph, which includes HAN [49], MAGNN [8], GTN [58], and so on. The stacking-fashion HGNNs adopt architectures consisting of stacking layers with the same structure, which is analog to homogeneous GNNs. Examples of these models include HGT [12], Simple-HGN [24], and the like.

Understanding Black-box Models. Though there exist many techniques for understanding black-box models [2, 4, 7, 9, 10, 18, 20, 39], the explainability of DGNs has recently received much attention. The pioneering research of explaining DGNs include GNNExplainer [54] and heat-map-based methods for GCNs [35], since then intensive research efforts have been devoted to explain homogeneous GNNs [11, 21, 23, 42, 47, 50, 54, 55, 57]. On the other hand, as an emerging line of works, there has not been any *model-agnostic* explainer for HGNNs to the best of our knowledge. Though an explainable fraud transaction detection model was proposed [36], which contains an initial attempt to study explainability on heterogeneous graphs, as it is fully based on a variant of GNNExplainer, we do not consider it an model-agnostic explainer for HGNNs.

HENCE-X *v.s.* PGM-Explainer. PGM-Explainer [47] is the most closely related work to our study, as both methods adopt the Bayesian network to model the non-linear dependency of DGN's prediction on the input. Although the authors claim PGM-Explainer guarantees the completeness of the output Markov blanket, it only selects top-K marginally dependent elements in the blanket basket, hence the guarantee is not guarded. In contrast, HENCE-X retains provable completeness of the Markov blanket, and ensures the output is a connected component, which is important and generally pursued by DGN explainers [21, 23, 50, 54, 57].

HENCE-X *v.s.* perturbation-based methods. Existing perturbation-based methods [7, 20] employ similar philosophy to find critical features, yet they rely on quantitative objects to measure the importance, while ignoring the causal effect between model's behavior and the input. On the contrary, HENCE-X employs the Bayesian network to capture the non-linear dependency of model prediction on the input element; meanwhile, HENCE-X guarantees the causal completeness of the output explanation.

6 CONCLUSION

Deep graph networks have been widely applied because of their outstanding performance. However human users cannot understand their decision making mechanism. Thus, explanation techniques for DGNs have been intensively desired. In this paper, we propose a heterogeneity-agnostic multi-level explainer named HENCE-X. HENCE-X is a causality-guided method that employs a Bayesian network to model the DGN's behavior, so that the Markov blanket of the prediction can be discovered as the explanation. Experiments on three real-world datasets have demonstrated the effectiveness of HENCE-X over the SOTA.

REFERENCES

[1] [n.d.]. Supplementary Materials. https://github.com/Gori-LV/HENCE-X.

[2] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.

[3] Mohit Bajaj, Lingyang Chu, Zi Yu Xue, Jian Pei, Lanjun Wang, Peter Cho-Ho Lam, and Yong Zhang. 2021. Robust counterfactual explanations on graph neural networks. *Advances in Neural Information Processing Systems* 34 (2021), 5644–5655.

[4] Samprit Chatterjee. 1988. *Sensitivity analysis in linear regression.* Wiley, New York.

[5] Anthony Costa Constantinou, Norman Fenton, William Marsh, and Lukasz Radlinski. 2016. From complex questionnaire and interviewing data to intelligent Bayesian network models for medical decision support. *Artificial intelligence in medicine* 67 (2016), 75–93.

[6] Asim Kumar Debnath, Rosa L. Lopez de Compadre, Gargi Debnath, Alan J. Shusterman, and Corwin Hansch. 1991. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry* 34, 2 (1991), 786–797.

[7] Ruth C Fong and Andrea Vedaldi. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision.* 3429–3437.

[8] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of The Web Conference 2020.* 2331–2341.

[9] Toshimitsu Homma and Andrea Saltelli. 1996. Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety* 52, 1 (1996), 1–17.

[10] Bingde Hu, Xingen Wang, Zunlei Feng, Jie Song, Ji Zhao, Mingli Song, and Xinyu Wang. 2022. HSDN: A High-Order Structural Semantic Disentangled Neural Network. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[11] Jinlong Hu, Tenghui Li, and Shoubin Dong. 2020. GCN-LRP explanation: exploring latent attention of graph convolutional networks. In *IJCNN.* IEEE, 1–8.

[12] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020.* 2704–2710.

[13] Shahab Wahhab Kareem, Farah Qasim Ahmed Alyousuf, Kosrat Ahmad, Roojwan Hawezi, and Hoshang Qasim Awla. 2022. Structure Learning of Bayesian Network: A Review. *QALAAI ZANIST JOURNAL* 7, 1 (2022), 956–975.

[14] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017.* OpenReview.net.

[15] Daphne Koller and Nir Friedman. 2009. *Probabilistic graphical models: principles and techniques.* MIT press.

[16] Agneza Krajna, Mihael Kovac, Mario Brcic, and Ana Sarcevic. 2022. Explainable Artificial Intelligence: An Updated Perspective. In *MIPRO.* IEEE, 859–864.

[17] Harold William Kuhn and Albert William Tucker. 1953. *Contributions to the Theory of Games.* Vol. 2. Princeton University Press.

[18] Ansong Li, Zhiyong Cheng, Fan Liu, Zan Gao, Weili Guan, and Yuxin Peng. 2022. Disentangled graph neural networks for session-based recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[19] Guangdi Li, Anne-Mieke Vandamme, and Jan Ramon. 2014. Learning ancestral polytrees. *Learning Tractable Probabilistic Models* (2014).

[20] Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220* (2016).

[21] Wanyu Lin, Hao Lan, and Baochun Li. 2021. Generative causal explanations for graph neural networks. In *International Conference on Machine Learning.* PMLR, 6666–6679.

[22] Ana Lucic, Maartje ter Hoeve, Gabriele Tolomei, Maarten de Rijke, and Fabrizio Silvestri. 2021. CF-GNNExplainer: Counterfactual Explanations for Graph Neural Networks. *arXiv preprint arXiv:2102.03322* (2021).

[23] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. 2020. Parameterized Explainer for Graph Neural Network. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020.*

[24] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are we really making much progress?: Revisiting, benchmarking and refining heterogeneous graph neural networks. In *KDD.* ACM, 1150–1160.

[25] Dimitris Margaritis and Sebastian Thrun. 1999. Bayesian network induction via local neighborhoods. *Advances in neural information processing systems* 12 (1999).

[26] John H McDonald. 2014. G-test of goodness-of-fit. *Handbook of biological statistics* 487 (2014), 53–58.

[27] John H McDonald. 2014. Small numbers in chi-square and G-tests. *Handbook of biological statistics* (2014), 86–89.

[28] Ayse Tosun Misirli and Ayse Basar Bener. 2014. Bayesian networks for evidence-based decision-making in software engineering. *IEEE Transactions on Software Engineering* 40, 6 (2014), 533–554.

[29] Martin Neil, Norman Fenton, and Lars Nielson. 2000. Building large-scale Bayesian networks. *The Knowledge Engineering Review* 15, 3 (2000), 257–284.

[30] M Ouerd, B John Oommen, and Stan Matwin. 2000. A formalism for building causal polytree structures using data distributions. In *International Symposium on Methodologies for Intelligent Systems.* Springer, 629–637.

[31] M Ouerd, B John Oommen, and Stan Matwin. 2004. A formal approach to using data distributions for building causal polytree structures. *Information Sciences* 168, 1-4 (2004), 111–132.

[32] Judea Pearl. 1988. Markov and Bayesian networks: Two graphical representations of probabilistic knowledge. *Probabilistic Reasoning in Intelligent Systems* (1988), 77–141.

[33] Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan kaufmann.

[34] Judea Pearl et al. 2000. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress* 19, 2 (2000).

[35] Phillip E. Pope, Soheil Kolouri, Mohammad Rostami, Charles E. Martin, and Heiko Hoffmann. 2019. Explainability Methods for Graph Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019.* Computer Vision Foundation / IEEE, 10772–10781.

[36] Susie Xi Rao, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang. 2021. xFraud: explainable fraud transaction detection. *Proceedings of the VLDB Endowment* 3 (2021), 427–436.

[37] George Rebane and Judea Pearl. 1988. The recovery of causal poly-trees from statistical data. *Int. J. Approx. Reason.* 2, 3 (1988), 341.

[38] John T. Roscoe. 1975. *Fundamental research statistics for the behavioral sciences.* Holt, Rinehart and Winston.

[39] Andrea Saltelli, Stefano Tarantola, and KP-S Chan. 1999. A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics* 41, 1 (1999), 39–56.

[40] Mauro Scanagatta, Antonio Salmerón, and Fabio Stella. 2019. A survey on Bayesian network structure learning from data. *Progress in Artificial Intelligence* 8, 4 (2019), 425–439.

[41] Uma Sekaran and Roger Bougie. 2016. *Research methods for business: A skill building approach.* john wiley & sons.

[42] Caihua Shan, Yifei Shen, Yao Zhang, Xiang Li, and Dongsheng Li. 2021. Reinforcement Learning Enhanced Explainer for Graph Neural Networks. *Advances in Neural Information Processing Systems* 34 (2021).

[43] Lu Sun and Mineichi Kudo. 2019. Multi-label classification by polytree-augmented classifier chains with label-dependent features. *Pattern Anal. Appl.* 22, 3 (2019), 1029–1049.

[44] Masami Takikawa and Bruce D'Ambrosio. 1999. Multiplicative Factorization of Noisy-Max. In *UAI.* Morgan Kaufmann, 622–630.

[45] Juntao Tan, Shijie Geng, Zuohui Fu, Yingqiang Ge, Shuyuan Xu, Yunqi Li, and Yongfeng Zhang. 2022. Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning. In *Proceedings of the ACM Web Conference 2022.* 1018–1027.

[46] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018.* OpenReview.net.

[47] Minh N. Vu and My T. Thai. 2020. PGM-Explainer: Probabilistic Graphical Model Explanations for Graph Neural Networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020.*

[48] Peter R de Waal and Linda C Gaag. 2007. Inference and learning in multi-dimensional Bayesian network classifiers. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty.* Springer, 501–511.

[49] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The World Wide Web Conference.* 2022–2032.

[50] Xiang Wang, Ying-Xin Wu, An Zhang, Xiangnan He, and Tat-Seng Chua. 2021. Towards Multi-Grained Explainability for Graph Neural Networks. In *NeurIPS.* 18446–18458.

[51] Xiang Wang, Yingxin Wu, An Zhang, Xiangnan He, and Tat-Seng Chua. 2021. Towards Multi-Grained Explainability for Graph Neural Networks. *Advances in Neural Information Processing Systems* 34 (2021).

[52] Xiang Wang, An Zhang, Xia Hu, Fuli Feng, Xiangnan He, Tat-Seng Chua, et al. 2022. Deconfounding to Explanation Evaluation in Graph Neural Networks. *arXiv preprint arXiv:2201.08802* (2022).

[53] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *ICLR*.

[54] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*.

[55] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. 2020. XGNN: Towards Model-Level Explanations of Graph Neural Networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, 2020*. ACM, 430–438.

[56] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. 2022. Explainability in graph neural networks: A taxonomic survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[57] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. 2020. On Explainability of Graph Neural Networks via Subgraph Explorations. In *ICML (Proceedings of Machine Learning Research)*.

[58] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. *Advances in Neural Information Processing Systems* 32 (2019), 11983–11993.

[59] Nevin Lianwen Zhang and David L. Poole. 1996. Exploiting Causal Independence in Bayesian Network Inference. *J. Artif. Intell. Res.* 5 (1996), 301–328.

7 SUPPLEMENTARY MATERIALS

7.1 Definitions

In this subsection presents the definitions [15] used in the paper.

Definition 7.1 (Blocked Paths [15]). *Given a Bayesian network, a path between two nodes X and Y is blocked by a set Z of nodes if*

1. *Either that path contains a node Z that is in Z and the connection at Z is either serial or diverging.*
2. *Or that the path contains a node W such that W and its descendants are not in Z and the connection at W is a converging connection.*

Definition 7.2 (d-separation [15]). *Two nodes X and Y are d-separated by a set Z if all paths between X and Y are blocked by Z.*

Definition 7.3 (I-map [15]). *A DAG $\mathcal{G}$ is an I-map of a distribution $\mathcal{P}$ over a set of random variables V, if for any three disjoint subsets of variables X, Y, and Z, Z d-separates X and Y in $\mathcal{G}$ implies $X \perp\!\!\!\perp_{P} Y|Z$.*

Definition 7.4 (D-map [15]). *A DAG $\mathcal{G}$ is a D-map of a distribution $\mathcal{P}$ over a set of random variables V, if for any three disjoint subsets of variables X, Y, and Z, $X \perp\!\!\!\perp_{P} Y|Z$ implies Z d-separates X and Y in $\mathcal{G}$.*

Definition 7.5 (perfect map [15]). *A DAG $\mathcal{G}$ is a perfect map of a distribution $\mathcal{P}$ if it is both an I-map and a D-map.*

7.2 Proofs

In this subsection we present proofs of lemmas and theorem in Section 3.

Lemma 3.3. *Let $\mathrm{MB}_{\mathcal{B}'}(\Phi_t)$ be a Markov blanket of Φ_t in the surrogate network $\mathcal{B}'$ that does not contain any contribution variables, denote $\mathcal{M}$ as the subset of $\mathrm{MB}_{\mathcal{B}'}(\Phi_t)$, which includes all feature variables only; mathematically,*

$$\mathcal{M} = \left\{F_{v_i}^{(j)} : \exists j \in [0..d_{\zeta(v_i)}] \text{ s.t. } F_{v_i}^{(j)} \in \mathrm{MB}_{\mathcal{B}'}(\Phi_t), \text{ for all } v_i \text{ in } G_t\right\},$$

and let $\mathcal{V}$ be the set of vertexes in the input graph that are associated with variables in $\mathrm{MB}_{\mathcal{B}'}(\Phi_t)$, i.e.

$$\mathcal{V} = \left\{v_i : \exists V_i \in \mathrm{MB}_{\mathcal{B}'}(\Phi_t)\right\}. \tag{9}$$

If for all $v_i \in \mathcal{V}$, there exists a subset $\mathcal{M}' \subseteq \mathcal{M} \backslash \mathcal{F}_{\phi,t}(v_i)$ such that

$$\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} F_{v_i}^{(j)} \Big| \mathcal{M}', \ \forall F_{v_i}^{(j)} \in \mathcal{K}_{\Phi_t}(v_i) \backslash \mathcal{F}_{\phi,t}(v_i), \tag{10}$$

where $\mathcal{F}_{\phi,t}(v_i) = \left\{F_{v'}^{(j)} \in \mathrm{MB}_{\mathcal{B}'}(\Phi_t) : v' = v_i, j \in [0..d_{\zeta(v_i)}]\right\}$ and $\mathcal{K}_{\Phi_t}(v_i)$ is defined in Equation (5), then $\mathcal{M}$ is a Markov blanket of Φ_t in $\mathcal{B}^*$.

Proof. The lemma can be proved by contradiction. Assume $\mathcal{M}$ is *not* a Markov blanket of Φ_t in $\mathcal{B}^*$, then $\mathcal{M}$ cannot d-separate Φ_t from all other variables in $\mathcal{B}^*$ that are not in $\mathcal{M}$, which means there exists at least one active path p connecting to Φ_t that is not blocked by $\mathcal{M}$. Since Φ_t is a leaf node in $\mathcal{B}^*$, p must contain a direct parent of Φ_t, denoted by $\bar{F}$. Note that $\bar{F}$ is not in $\mathcal{M}$ as assumed. Since $\bar{F}$ is a direct parent of Φ_t in $\mathcal{B}^*$, it is connected to Φ_t in $\mathcal{B}'$ by a contribution variable $\bar{\xi}$ and a synthetic variable $\bar{V}$ according to the rules of constructing $\mathcal{B}'$, then $\bar{V}$ is a direct parent of Φ_t in $\mathcal{B}'$ and $\bar{V}$ must be included in $\mathrm{MB}_{\mathcal{B}'}(\Phi_t)$ because direct parents are always included in the Markov blanket [33]. Hence the vertex $\bar{v}$ associated with $\bar{V}$ is included in $\mathcal{V}$ and there exists a subset $\mathcal{M}' \subseteq \mathcal{M} \backslash \mathcal{F}_{\phi,t}(\bar{v})$ such that $\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} \bar{F} \big| \mathcal{M}'$ since $\bar{F} \in \mathcal{K}_{\Phi_t}(\bar{v}) \backslash \mathcal{F}_{\phi,t}(\bar{v})$, which means $\mathcal{M}'$ d-separates $\bar{F}$ and Φ_t in $\mathcal{B}'$, namely, all paths from $\bar{F}$ to Φ_t are blocked by $\mathcal{M}'$. However, there is a path from $\bar{F}$ to $\Phi_t : \bar{F} \to \bar{\xi} \to \bar{V} \to \Phi_t$, while $\mathcal{M}$ does not contain any contribution variable ξ_i^j or synthetic variable V_i, hence the above path can never be blocked by any subset of $\mathcal{M}$, which means there does not exist a subset $\mathcal{M}' \subseteq \mathcal{M}$ that can d-separate $\bar{F}$ from Φ_t. At this point, we reach a contradiction. In conclusion, there does not exists such an active path p in $\mathcal{B}^*$ that is not blocked by $\mathcal{M}$, and $\mathcal{M}$ is a Markov blanket of Φ_t in $\mathcal{B}^*$. □

Lemma 3.4. *In the surrogate network $\mathcal{B}'$, for any synthetic variable V_i, let Z be a set of variables that does not contain any of its grandparent feature variable $F_{v_i}^{(\cdot)}$'s or contribution variables, if there exists some $F_{v_i}^{(j)}$ such that $F_{v_i}^{(j)} \not\perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$, then $V_i \not\perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$.*

Proof. Since $F_{v_i}^{(j)} \not\perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$, then Z cannot d-separate $F_{v_i}^{(j)}$ from Φ_t, there exists at least one active path p in $\mathcal{B}'$ connecting $F_{v_i}^{(j)}$ and Φ_t that is not blocked by Z, which is in either of the below cases:

(1) p connects $F_{v_i}^{(j)}$ to Φ_t through $F_{v_i}^{(j)} \to \xi_i^j \to V_i$; or

(2) p connects $F_{v_i}^{(j)}$ to Φ_t through some parent node of $F_{v_i}^{(j)}$.

In case (1), V_i is obviously not blocked from Φ_t as it is contained in p. In case (2), V_i is also connected to Φ_t through a path p' formed by concatenating p with $F_{v_i}^{(j)} \to \xi_i^j \to V_i$. As p is connect to $F_{v_i}^{(j)}$ though its parent, $F_{v_i}^{(j)} \to \xi_i^j \to V_i$ and p have a junction at $F_{v_i}^{(j)}$ with *chain* pattern, *i.e.* it is a serial path and it has no node contained in Z, hence p' extends p to connect V_i while remaining an active path that is not blocked by Z [34]. In conclusion, if $F_{v_i}^{(j)} \not\perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$, then there exists an active path in $\mathcal{B}'$ connecting V_i and Φ_t that is not blocked by Z, thus $V_i \not\perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$. □

To illustrate the two cases in the proof, take V_2 in Figure 2(d) as an example. For case (1), p can be **P2**-“*DB2*”$\to \xi_2^2 \to V_2 \to$ **A1**-“*DB*” $\to \xi_4^1 \to V_4 \to \Phi_t$, then V_2 is connect to Φ_t. For case (2), p' can be $V_2 \leftarrow \xi_2^2 \leftarrow$ **P2**-“*DB2*” $\leftarrow V_1 \to$ **P3**-“*DB2*” $\to \xi_3^4 \to V_3 \to \Phi_t$; note that at V_1, the junction pattern is *fork*, where p' is not blocked as long as V_1 is not in Z [34].

Lemma 3.5. *In the surrogate network $\mathcal{B}'$, for any synthetic variable V_i, let Z be a set of variables that does not contain any of its grandparent feature variable $F_{v_i}^{(\cdot)}$'s or contribution variables, if $V_i \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$, then all its grandparent feature variable and parent contribution variables are independent from the prediction variable conditioned on Z, namely, $F_{v_i}^{(j)} \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$ and $\xi_i^j \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$ for all $j \in [0..d_{\zeta(v_i)}]$.*

Proof. The *converse-negative proposition* of Lemma 3.4 shows that under the stated condition, *if $V_i \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$, then $F_{v_i}^{(j)} \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z, \forall F_{v_i}^{(j)} \in \mathcal{K}_{\Phi_t}(v_i)$*. Hence, to prove Lemma 3.5, it suffices to prove that all parent contribution variables of V_i are independent from the prediction variable conditioned on Z, *i.e.* $\xi_i^j \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \big| Z$. According to the building rules of the surrogate network, for any synthetic variable in $\mathcal{B}'$, all its parent contribution ξ_i^j are contained in one serial path *only*, that is $V_{i'} \rightarrow F^{(j)}_{v_i} \rightarrow \xi^j_i \rightarrow V_i \rightarrow F^{(j')}_{v_{i''}}$. If V_i is d-separated from from Φ_t by Z, then the above path is blocked by Z, which means ξ^j_i is d-separated from Φ_t by Z, hence $\xi^j_i \perp\!\!\!\perp_{\mathcal{B}'} \Phi_t \Big| Z$. □

Lemma 3.6. *Given a pretrained DGN ϕ and a target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_\phi(t)$ and every necessary vertex in G_t is connected to the target node through other necessary vertex(es). Let $\mathcal{E}'$ be a set of vertexes, whose induced subgraph of G_t is a connected component that contains the target t, denote its neighborhood as $\boldsymbol{Ne}(\mathcal{E}') = \bigcup_{v \in \mathcal{E}'} Adj[v] \Big\backslash \mathcal{E}'$, where $Adj[v]$ is the set of all adjacent neighbor(s) of the vertex v; and denote the set of synthetic variables and feature variables induced by $\mathcal{E}'$ as*

$$\begin{aligned} \mathcal{Z}(\mathcal{E}') =& \Big\{ V_i : v_i \in \mathcal{E}' \cup \boldsymbol{Ne}(\mathcal{E}') \Big\} \\ & \bigcup \Big\{ F^{(j)}_{v'} \in \mathcal{K}_{\Phi_t}(v') : v' \in \mathcal{E}' \cup \boldsymbol{Ne}(\mathcal{E}') \Big\}. \end{aligned} \tag{11}$$

If for all $V' \in \{V' : v' \in \boldsymbol{Ne}(\mathcal{E}')\}$, there exists a subset $\mathcal{M}' \subseteq \mathcal{Z}(\mathcal{E}')$ such that $\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} V' \Big| \mathcal{M}'$, or $\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} F^{(j)}_{v_i} \Big| \mathcal{M}'$ for all grandparent feature variables of V', then a Markov blanket of Φ_t can be found as a subset of $\mathcal{Z}(\mathcal{E}')$.

Proof. Under the stated condition of this lemma, $\mathcal{E}'$ separates all random variables in $\mathcal{P}_{\Phi_t}(t)$ into two groups based on the topology of the input graph. To prove $\mathcal{Z}$ contains a complete Markov blanket, one needs to firstly prove all contribution variables related to vertexes in $\mathcal{E}'$ are conditionally independent from Φ_t; secondly, all random variables that are not related to vertexes in $\mathcal{E}'$ are also conditionally independent from Φ_t. The former is obvious as, for any contribution variable ξ^j_i, there is only one serial path that contains it (see proof of Lemma 3.5), thus the path can always be blocked by any set that contains its adjacent neighbors. The second statement can be proved by contradiction. Assume $\mathcal{Z}$ does not contain a complete Markov blanket of Φ_t, then there exists at least one necessary variable $\bar{V}$ outside $\mathcal{Z}$. It can be either

(1) associated with a vertex $\bar{v}$ in $\boldsymbol{Ne}(\mathcal{E}')$; or
(2) associated with a vertex $\bar{v}$ outside $\boldsymbol{Ne}(\mathcal{E}') \bigcup \mathcal{E}'$.

In case (1), there exists a set $\mathcal{M}' \subseteq \mathcal{Z}$ such that (*a*) $\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} V' \Big| \mathcal{M}'$, then $\bar{V}$ can be excluded from a Markov blanket, which contradicts with the assumption of $\bar{V}$ being a necessary variable; or (*b*) $\Phi_t \perp\!\!\!\perp_{\mathcal{B}'} F^{(j)}_{\bar{v}} \Big| \mathcal{M}'$ for all grandparent feature variables of $\bar{V}$, then every $F^{(j)}_{\bar{v}}$ is d-separated by $\mathcal{M}'$ from Φ_t, thus $\bar{V}$ must be d-separated by $\mathcal{M}'$ from Φ_t as well, otherwise $F^{(j)}_{\bar{v}}$'s can be connected to Φ_t through $\bar{V}$ (see Proof. of Lemma 3.4), hence $\bar{V}$ can be excluded from a Markov blanket and contradicts with the assumption again. In case (2), because all necessary vertex is connected to the target by other necessary ones, then there must exist a path in the input graph composed of necessary vertexes that connects $\bar{v}$ to t, hence there must exist some necessary variable in $\boldsymbol{Ne}(\mathcal{E}')$, which is exactly case (1). Combining both cases, we prove that there does not exist a necessary variable outside $\mathcal{Z}$, then all synthetic variables outside $\mathcal{Z}$ are independent from Φ_t conditioned on some subset of $\mathcal{Z}$. According to Lemma 3.5, all their parent contribution variables and grandparent feature variables are independent from Φ_t conditioned on the same set. In conclusion, any variable outside $\mathcal{Z}$ is conditionally independent from Φ_t. To summarize, we have proved all contribution variables related to vertexes in $\mathcal{E}'$ and all other variables that are *not* related to vertexes in $\mathcal{E}'$ are both conditionally independent from Φ_t, and one can find a Markov blanket of Φ_t in $\mathcal{Z}$ under the stated condition. □

Theorem 3.7. Given a pretrained DGN ϕ and a target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_\phi(t)$ and every necessary vertex in G_t is connected to the target node through other necessary vertex(es), running Algorithm 1 to solve the heterogeneity-agnostic multi-level explanation generation problem defined in Definition 2.3 outputs the topological-level explanation $\mathcal{E}_{topo}$ and the feature-level explanation $\mathcal{E}^v_{feat}$'s for each vertex v in $\mathcal{E}_{topo}$. Let $\mathcal{M}^* = \Big\{ F^{(j)}_{v_i} : f^{(j)}_{v_i} \in \bigcup_{v' \in \mathcal{E}_{topo}} \mathcal{E}^{v'}_{feat} \Big\}$, then $\mathcal{M}^*$ is a Markov blanket of Φ_t in $\mathcal{B}^*$.

Proof. To prove the theorem, we first prove the ***growing phase*** (*line 7-19*) produces a Markov blanket of Φ_t in the surrogate network $\mathcal{B}'$. We claim that the union set $\{\hat{V}\text{'s}\} \bigcup \mathcal{M}$ is a Markov blanket of Φ_t in $\mathcal{B}'$ upon finishing the *while* loop, where $\{\hat{V}\text{'s}\}$ is the set of all $\hat{V}$'s identified as the most promising synthetic variable (*line 8*) in the loop and $\mathcal{M}$ is a set made of feature variables only. Notice the stop condition of the loop states that every synthetic variable in the neighborhood of $\mathcal{E}_{topo}$ (enclosed in $\mathcal{U}$) is tested independent from Φ_t or so are all of its grandparent feature variables (the synthetic variable is added to $\mathcal{I}$). Then taking $\mathcal{E}_{topo}$ (after the *while* loop ends) as $\mathcal{E}'$ in Lemma 3.6, a Markov blanket of Φ_t is guaranteed in $\mathcal{Z}(\mathcal{E}_{topo})$ as defined in Equation (11). During the execution of the loop, every variable in $\mathcal{Z}(\mathcal{E}_{topo})$ has been tested whether it is conditionally independent from Φ_t except in three groups:

(1) grandparent feature variables of synthetic variables that are conditionally independent from Φ_t.
(2) parent contribution variables of synthetic variables that are conditionally independent from Φ_t.
(3) other contribution variables.

For the tested variables, those that are not conditionally independent are then added to the growing blanket, hence the property of the Markov blanket is ensured. For untested variables, the first two groups are proved to be out of a Markov blanket by Lemma 3.5. For the last group, those contribution variables' parent feature variables are immediately tested after their common child synthetic variable, by whom the only path containing the contribution variable can be well-blocked (see proof of Lemma 3.5). Hence, in summary, $\{\hat{V}\text{'s}\} \bigcup \mathcal{M}$ produced by the growing phase is a Markov blanket of Φ_t in $\mathcal{B}'$. In the ***shrinking phase*** (*line 20-22*), if a vertex in $\mathcal{E}_{topo}$ is associated with a synthetic variable that is conditionally independent from Φ_t, all of its grandparent feature variables are removed from $\mathcal{M}$, because they are also conditionally independent from Φ_t according to Lemma 3.5. Thus after shrinking, $\{\hat{V}\text{'s}\} \bigcup \mathcal{M}$ remains a Markov blanket of Φ_t in $\mathcal{B}'$.

Finally, let $\mathcal{M}^*$ in this theorem be the set $\mathcal{M}$ in Lemma 3.3 and $\mathcal{E}_{topo}$ be the set $\mathcal{V}$ defined in Equation (9), by taking $\mathcal{M}$ in the iteration of adding the vertex v (*line 11*) as $\mathcal{M}'$ for every synthetic variable in $\{\hat{V}\text{'s}\}$, Equation (10) is well satisfied. According to Lemma 3.3, $\mathcal{M}^*$ is a Markov blanket in $\mathcal{B}^*$, the and theorem is proved. □

Method	DBLP	IMDB	MUTAG
PGExplainer	38480.20s	4376.92s	51.94s
Gem	13160.22s	1288.88s	9.23s
RG-Explainer	46176.24s	7003.52s	467.46s

Table 6: Training time for inductive explainers

Theorem 3.8. *Given a pretrained DGN ϕ and a target node t, assume there exists a perfect map $\mathcal{B}^*$ of the model behavior distribution $\mathcal{P}_{\phi}(t)$ and every necessary vertex in G_t is connected to the target node through other necessary vertex(es), running Algorithm 1 to solve the heterogeneity-agnostic multi-level explanation generation problem defined in Definition 2.3 outputs the topological-level explanation $\mathcal{E}_{topo}$. Let $\mathcal{R} = \{V_i : \exists v_i \in \mathcal{E}_{topo}\}$, then $\mathcal{R}$ is the set of all direct parents of Φ_t in $\mathcal{B}'$, i.e. $\mathcal{R} = \boldsymbol{Pa}(\Phi_t)$.*

Proof. We first prove (i) $\boldsymbol{Pa}(\Phi_t) \subseteq \mathcal{R}$. It has been proven that $\{\hat{V}\text{'s}\} \bigcup \mathcal{M}$ is a Markov blanket of Φ_t in $\mathcal{B}'$ in Theorem 3.7, then it must contain all direct parents of Φ_t by definition (direct parents are always included in Markov blanket). $\mathcal{R}$ differ from $\{\hat{V}\text{'s}\}$ in that it does not include those synthetic variables whose grandparent feature variables are conditionally independent from Φ_t, which means these feature variables are not direct parents of Φ_t in the original perfect map $\mathcal{B}^*$, then the corresponding synthetic variables are not direct parents of Φ_t in $\mathcal{B}'$ either according to the building rule of surrogate network $\mathcal{B}'$. Next, we prove (ii) $\mathcal{R} \subseteq \boldsymbol{Pa}(\Phi_t)$ by contradiction, if there exist a variable in $\mathcal{R}$ that is not in $\boldsymbol{Pa}(\Phi_t)$, denote the variable $\bar{V}$, then $\bar{V}$ is not a direct parent of Φ_t in $\mathcal{B}'$, then all its grandparent feature variables are not direct parent of Φ_t in $\mathcal{B}^*$. Hence, $\bar{V}$ can be d-separated from Φ_t by some subset of $\mathcal{M}$, which contradict with *line 21* in the algorithm. Combining (i) and (ii), we prove that $\mathcal{R} = \boldsymbol{Pa}(\Phi_t)$. □

7.3 Training time for inductive explainers