We would like to thank reviewers for the constructive feedback and the positive recommendation. The reference numbers used here are from the manuscript. Following are our responses to the concerns raised by reviewers.

(R1) The choice of graph perturbation method. There is a trade-off in the amount of perturbation applied on the input graph: too large perturbation makes the explainer fail to capture the local behavior of the model at the input, while too small perturbation prevents the explainer from efficiently observing the contributions of graph's components toward the model's output. While zeroing-out features leads to abrupt change in the model's output, randomizing features requires a large number of samples for stable explanations. We observe that averaging features among nodes lead to better performance for PGM-Explainer. As stated at line 140, this method is used in our implementation.

(R2) $\perp\!\!\!\perp$ symbol in Alg.1. Thank you for pointing this out! It should be $\not\perp\!\!\!\perp$ which indicates independent relationship.

(R2) 4 nodes in Fig. 5. The 4 nodes highlighted in Fig. 5 of PGM-Explainer is not a mistake. In this experiment, we do not restrict the number of nodes returned by PGM-Explainer. In fact, they are included as long as they are in the Markov-Blanket of the target variable. From that, we obtain explanations containing either $2, 3$, or 4 nodes with an average of 3.08. Since other methods such as SHAP and GRAD do not have equivalent mechanism of selecting features, we set their number of returned nodes to 3. We will add this description to the final version.

(R2) Goals of Synthetic dataset and the number of nodes k in explanations. The experiments are designed by [23] so that we can identify which graph's components are constituted to the GNNs' predictions. Specifically, the reason for a node's role should be a set of nodes/features in the corresponding motif. We agree with **R2** that some nodes in the motif might not be exploited by the GNNs to compute the predictions; however, the number of returned nodes k is set to be the number of nodes in the motif because of the original experimental settings [23]. In practice, PGM-Explainer can automatically select highly contributed nodes as the Markov-Blanket of the target variable.

(R3) Why chose GCN [5]. We chose it because it is one of the most well-known graph neural networks and relatively fast to train [32].

(R3) Dependency of PGM. Although we do not experimentally show the accuracy and precision gains of PGM-Explainer are directly from exploiting the dependency among features, we theoretically prove all dependent variables must be included in the explanation (Theorem 1 and 2). Furthermore, with sufficient number of samples n, all dependency statements in $\mathcal{D}_t$ must be included in the explanation graphical model.

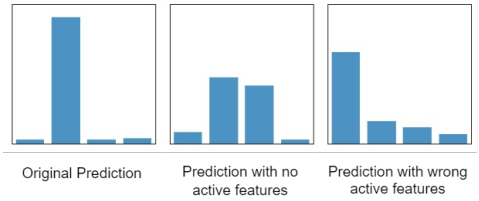

Figure A: Predictions of GNN in experiment on Fig.1 for the central-red node with different activated features.

(R3) Mutual information (MI) objective (line 302-303). Since the MI objective does not maximize the predicted label, features promoting other classes can also maximize the objective. We demonstrate this in the same GNN considered in Fig.1 of the manuscript. Instead of considering node E, we examine the prediction *red* on the central node A. Fig. A shows the soft-max predictions of A when different graph's components are activated/inactivated. By only activating the yellow node (shown in Fig.1(d) in the manuscript), which is not a node in the motif, the conditional entropy of A's soft-max output reduced significantly (from the middle to the right figure of Fig. A). However, this node does not contribute constructively to the original prediction (the second class). We observe that this phenomenon occurs more frequently with nodes of high degree. We will smooth this expression in the final version.

(R4) Experimental setup and Time complexity. The models are directly taken from previous work [23] and [32] (stated at line 250 and 267). Specifically, all models are trained using Adam optimizer. In the node classifications, all models have 3 graph layers with the number of parameters between 1102 and 1548. The train/validation/test split is $80/10/10\%$. The models are trained for 1000 epochs with learning rate 0.001. For the graph classification, the dataset is divided into 55K/5K/10K. The model has 101365 parameters and converges at epoch 188 with learning rate is automatically chosen between $\{10^{-3}, 10^{-4}\}$. The models' accuracy and the number of samples n used by PGM-Explainer to generate the explanations are shown in the below table. Here, n is chosen based on the expected number of variables contributing to the target prediction. If the graph maximum degree is much less than the number of nodes, the time complexity of PGM-Explainer is dominated by n forwarding computations of the model in the data generation step. We will include this discussion in the final version.

	Node Classification								**Graph Classification**
Experiment	syn1	syn2	syn3	syn4	syn5	syn6	Btc-alpha	Btc-OTC	GCN-MNIST
Accuracy	97.9	85.4	100.0	99.4	89.1	99.3	93.9	89.5	90.4
No. Samples n	800	800	800	1600	4000	800	1000	1000	400

Finally, we thank reviewers for constructive suggestions in including some formal definitions (**R1**: Barabasi-Albert graph, Perfect map, Markov-Blanket...) and illustrations (**R2**: PGM with/without child constraint). We really appreciate these suggestions and, if our work is accepted, we will include them in the final version.