Improving Subgraph Recognition with Variational Graph Information Bottleneck

Junchi Yu[1,2], Jie Cao[1], Ran He[1,2,3*]

[1]NLPR&CRIPAC Institute of Automation, Chinese Academy of Sciences, China
[2]University of Chinese Academy of Sciences, China
[3]Center for Excellence in Brain Science and Intelligence Technology, CAS, China

yujunchi2019@ia.ac.cn, jie.cao@cripac.ia.ac.cn, rhe@nlpr.ia.ac.cn

Abstract

Subgraph recognition aims at discovering a compressed substructure of a graph that is most informative to the graph property. It can be formulated by optimizing Graph Information Bottleneck (GIB) with a mutual information estimator. However, GIB suffers from training instability and degenerated results due to its intrinsic optimization process. To tackle these issues, we reformulate the subgraph recognition problem into two steps: graph perturbation and subgraph selection, leading to a novel Variational Graph Information Bottleneck (VGIB) framework. VGIB first employs the noise injection to modulate the information flow from the input graph to the perturbed graph. Then, the perturbed graph is encouraged to be informative to the graph property. VGIB further obtains the desired subgraph by filtering out the noise in the perturbed graph. With the customized noise prior for each input, the VGIB objective is endowed with a tractable variational upper bound, leading to a superior empirical performance as well as theoretical properties. Extensive experiments on graph interpretation, explainability of Graph Neural Networks, and graph classification show that VGIB finds better subgraphs than existing methods [1].

1. Introduction

Graph classification, which aims to identify the labels of graph-structured data, has attracted much attention in diverse fields such as biochemistry [11, 19, 20, 34], social network analysis [14, 22, 44], and computer vision [7, 23, 24, 27]. Recently, there has been a surge of interest

---

[*]Corresponding Author
[1]Code is avaliable on https://github.com/Samyu0304/VGIB

in its reverse problem. That is, to recognize a compressed subgraph of the input, which is most predictive to the graph label [52]. Such a subgraph enjoys superior property for predictive performance since it drops noisy and redundant information and only preserves label-relevant information [46,53]. Meanwhile, the produced subgraph serves as an intrinsic explanation to the prediction of the graph model [49]. Hence, recognizing a compressed yet informative subgraph, namely the subgraph recognition, is the fundamental problem of many tasks. For example, biochemists are interested in discovering the substructure of the molecule which most affects the molecule properties [20, 53]. In the explainability of Graph Neural Networks (GNNs), it is vital to generate the explanatory subgraph of the input, which faithfully interprets the predicted results [28, 49]. In graph classification, the significant substructures, such as nodes, edges, and subgraphs, are highlighted to improve the predictive performances [3, 26, 39]. The subgraph recognition problem is first studied in a unified view under the Graph Information Bottleneck (GIB) framework [52]. It employs the Shannon mutual information to quantify the compressed and informative nature of the subgraph distribution. Although GIB permits theoretical analysis of the subgraph recognition problem, its optimization process is inefficient and unstable due to mutual information estimation, which is shown in Fig. 1. Meanwhile, the inaccurate estimated value leads to degenerated performance on subgraph recognition. These issues motivate us to advance the existing framework for improved subgraph recognition.

In this work, we address all the above issues with the proposed Variational Graph Information Bottleneck (VGIB). VGIB reformulates the subgraph recognition problem into two steps: graph perturbation and subgraph selection. During graph perturbation, VGIB employs a noise injection method to selectively inject noises into the input graph to obtain the perturbed graph. The intuition is that the noise
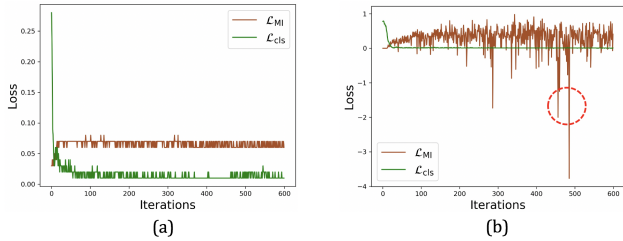
Figure 1. Training dynamics of VGIB and GIB. $\mathcal{L}_{\text{cls}}$ and $\mathcal{L}_{\text{MI}}$ refer to the prediction and compression term of two methods. (a). VGIB converges fast and stable. (b). GIB suffers from an unstable training process and inaccurate estimation of mutual information in the red circle (mutual information is non-negative).

injection naturally modulates the information flow from the input graph to the perturbed graph. Specifically, the more injected noise leads to more significant information distortion in the perturbed graph, which in analogy to the compressed nature of the subgraph. Hence, VGIB approaches the compression condition in GIB with the total amount of injected noise in the perturbed graph, leading to a tractable *variational* upper bound. Meanwhile, VGIB encourages the perturbed graph to be informative of the graph property, which indicates less injected noise. This trade-off condition guides the noise injection module to only inject noise into the insignificant substructure while preserving the predictive portion of the input. However, the design of noise injection is non-trivial. First, the action space of noise injection is discrete, leading to the difficulty of optimization with gradient methods. Hence, we employ the Gumbel-Softmax reparametrization for noise injection. Secondly, injecting random noise will break the semantic information of the input and lead to the difficulty of noise quantification. To this end, we customize the Gaussian prior for each input graph. With the above configurations, the VGIB objective enables a tractable variational upper bound and is efficient to optimize the gradient-based method. After training VGIB with graph perturbation, only the insignificant substructure of the graph is perturbed and the informative subgraph is well-preserved. Thus, in the subgraph selection step, one can obtain the found subgraph by dropping the injected noise.

We evaluate the proposed VGIB framework on various tasks, including explainability of GNNs, graph interpretation, and graph classification. The experimental results show that VGIB enjoys significant efficiency in optimization and outperforms the baseline methods with better found subgraphs.

2. Related Work

Information Bottleneck. The information bottleneck (IB) principle attempts to juice out a compressed but predictive code of the input signal [43]. Alemi et al. [2] first empowers deep learning with a variational information bottleneck (VIB). Currently, the applications of IB and VIB in

deep learning are mainly attributed to representation learning and feature selection. In the representation learning scenario, researchers employ a deterministic or stochastic encoder to learn a compressed yet meaningful representation of the input data, to facilitate various downstream tasks, such as computer vision [29, 30], reinforcement learning [13, 16], natural language processing [45], speech and acoustics [32], and node representation learning [46]. For the feature selection, IB is used to select a subset of input features such as pixels in images or dimensions in vectors, which are maximally predictive to the label of input data [1, 21, 38]. [1, 38] inject noises into the intermediate representations of a pretrained network and select the areas with maximal information per dimension. [21] learns the drop rates for each dimension of the vector-structured features. Unlike the prior work on the regular data, Yu et al. [52] first recognize a predictive yet compressed subgraph from the irregular graph input and thus facilitates various graph-level tasks.

Graph Classification. The goal of graph classification is to infer the label or property of an input graph. Recently, there is a surge of interest in applying the Graph Neural Network (GNN) for graph classification [33, 56]. It first aggregates the messages in the neighborhoods for node representations, which are pooled for the graph representations for prediction by a readout function. The typical implementations of readout are mean and sum functions [15, 22, 44, 47]. Besides, it is popular to leverage the hierarchical and more complex information in the graph, which leads to the graph-pooling methods [5, 25, 33, 50, 56]. These methods generally leverage all the information in graphs for prediction, neglecting the importance of the informative substructure. Hence, subgraph recognition can enhance graph classification with the label-relevant information in the graphs [52].

Subgraph Discovery. Subgraph discovery in the literature of traditional data mining refers to discovering subgraphs with specific topology [9, 12, 48]. It is similar to the data generation task [40, 51]. Recently, there is a trend to leverage the importance of subgraphs in graph learning. At node-level tasks, researchers focus on passing the message of a neighborhood subgraph to the central node [8, 14, 15]. NeuralSparse [57] chooses the most relevant K neighborhoods of a central node for robust node classification. At the graph level, it is popular to discover the information in subgraphs for learning graph representations. Infograph [39] maximize the mutual information between representations of graphs and the corresponding local patches. Another direction closely related to subgraph recognition is to interpret a pretrained GCN with interpretable subgraphs. GNNExplainer [49] discover the neighborhood subgraph, which maximally affects the prediction of the central node. SubgraphX [55] explains the prediction of GCN with a subgraph found by Monte Carlo Tree Search. However, these

methods only explore the subgraph recognition problem with specific tasks, and thus lack an unified view of the subgraph recognition problem.

3. Notations and Backgrounds

In this section, we introduce our notations and preliminaries. Let $G = \{A, X\} \in \mathbb{G}$ be a graph with n nodes, with $A \in \mathbb{R}^{n \times n}$ and $X \in \mathbb{R}^{n \times d}$ being its adjacent matrix and node feature matrix. We denote $\{(G_1, Y_1), (G_2, Y_2), \cdots, (G_n, Y_n)\}$ as the set of n graphs with its corresponding categorical labels or real-value properties. We use G_{sub} to denote the subgraph in G. We denote $I(X, Y)$ as the mutual information between the random variables X and Y, which takes the form:

$$I(X, Y) = \int_X \int_Y p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \mathrm{d}x \mathrm{d}y$$

3.1. Graph Information Bottleneck

Given the input data X and its label Y, the information bottleneck (IB) [43] principle learns the *minimal sufficient* representation Z by optimizing the IB objective: $\min_Z -I(Z, Y) + \beta I(Z, X)$. Here β is the Lagrangian parameter to balance the two terms. Inspired by it, Yu et al. propose Graph Information Bottleneck (GIB) principle to recognize an informative yet compressed subgraph from the original graph [52]. The GIB objective is as follows:

$$\min_{G_{sub}} -I(G_{sub}, Y) + \beta I(G_{sub}, G). \tag{1}$$

The first term in Eq. 1 encourages G_{sub} to be informative to the graph label Y. And the second term minimizes the mutual information of G and G_{sub} so that G_{sub} only receives limited information from the input graph G. The subgraph found by GIB is denoted as the **IB-subgraph**: $G_{sub}^* = \arg\min_{G_{sub}} -I(G_{sub}, Y) + \beta I(G_{sub}, G)$. The GIB objective cannot be directly optimized since the mutual information is intractable to compute. Hence, it first estimates the mutual information with MINE [4], and use the estimated value as a proxy in the optimization of GIB. This bilevel process is inefficient in optimization since it is time-consuming to estimate the mutual information. Meanwhile, inaccurate estimation also leads to an unstable training process and degenerated results.

4. Method

4.1. Compression via Noise Injection

Rather than directly evaluate the compression quality of G_{sub} with $I(G, G_{sub})$, we inject noise into the node representations of G for an alternative as shown in Fig. 2.

For an input graph $G \in \mathbb{G}$ with node feature matrix X, adjacent matrix A and degree matrix D, we first generate the node representations with a l-layer GNN:

$$\begin{aligned} H &= \mathrm{GNN}(A, X; W^1, \cdots, W^l) \\ &= \underbrace{\sigma(D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \cdots \sigma(D^{-\frac{1}{2}} A D^{-\frac{1}{2}} X W^1) \cdots W^l)}_{l-\text{layer}} \end{aligned}$$
(2)

where $H = [h_1, h_2, \cdots, h_n]^T$ is the node representation matrix. $W^1, \cdots, W^l$ are the parameters at different layers.

Then we damp the information in G by injecting noises into node representations with a learned probability. Let ϵ be the noise sampled from a parametric noise distribution. We assign each node a probability of being replaced by ϵ. Specifically, for the i-th node, we learn the probability p_i with a Multi-layer Perceptron (MLP). Then, we add a Sigmoid function on the output of MLP to ensure $p_i \in [0, 1]$:

$$p_i = \mathrm{Sigmoid}(\mathrm{MLP}(h_i)). \tag{3}$$

We then replace the node representation h_i by ϵ with probability p_i:

$$z_i = \lambda_i h_i + (1 - \lambda_i)\epsilon, \tag{4}$$

where $\lambda_i \sim \mathrm{Bernoulli}(p_i)$. The transmission probability p_i controls the information sent from h_i to z_i. If $p_i = 1$, then all the information in h_i are transfered to z_i without loss. On the contrary, when $p_i = 0$, then z_i contains no information from h_i but only noise. Compared with dropping nodes for compression in GIB, this method allows to flexibly adjust the amount of information from h_i to z_i by changing p_i. We denote the $G_N = \{A, Z\}$ as the perturbed graph. At graph level, the transmissioin probability of each node determines the information sent from G to G_N in a similar way. Therefore, we can compress the information of G into G_N with a set of p_i. We hope p_i is learnable so that we can selectively preserve the information in G_N. However, λ_i is a discrete random variable and we can not directly calculate the gradient of p_i. Therefore, we employ the concrete relaxation [10, 18] for λ_i:

$$\hat{\lambda}_i = \mathrm{Sigmoid}(\frac{1}{t} \log \frac{p_i}{1 - p_i} + \log \frac{u}{1 - u}), \tag{5}$$

where t is the temperature parameter and $u \sim \mathrm{Uniform}(0, 1)$. Another key component in the noise injection is the specification of the injected noise. Notice the arbitrary noise is detrimental to the semantic of the input graph. This will lead the prediction of the perturbed graph to be different from the graph property. Moreover, appropriately selected noise will endow the whole objective with a variational upper bound. Please refer to Section 4.3 for detailed discussions.

4.2. Variational Graph Information Bottleneck

The perturbed graph G_N is employed to distill the actionable information of G for predicting its label Y. Specifically, we compress the information of G via noise injection
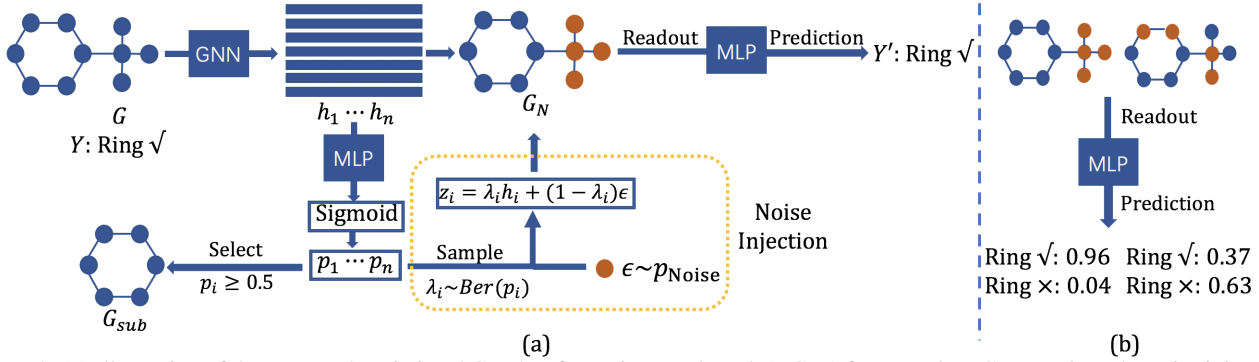
Figure 2. (a) Illustration of the proposed Variational Graph Information Bottleneck (VGIB) framework. VGIB employs the noise injection method to generate a perturbed graph G_N, which is used as a "bottleneck" to distill the actionable information for predicting the graph label. The objective of VGIB has a tractable upper bound that is easy to optimize. (b) Intuition of the noise injection method. perturbing a significant subgraph is more harmful to the graph label than perturbing a label-irrelevant subgraph.

to obtain G_N. Meanwhile, we hope that G_N is maximally informative to Y, which leads to a novel Variational Graph Information Bottleneck (VGIB) framework:

$$\min_{G_N} -I(G_N, Y) + \beta I(G_N, G). \qquad (6)$$

The first term encourages G_N to be sufficient for predicting the graph label Y, and the second term constrains the information that G_N receives from G. These two terms require us to inject noise into G selectively so that G_N receives actionable information as much as possible. The intuition is that injecting noises into the IB-subgraph of G is more harmful to the functionality of G than that into the label-irrelevant substructures. In that sense, the nodes in the IB-subgraph are less likely to be injected with noise. Therefore, we can select the IB-subgraph from G_N by this criterion after training VGIB. We introduce the following lemma before justifying the above formulation.

Lemma 4.1. Let $G \in \mathbb{G}$ and $Y \in \mathbb{R}$ be the graph and its label. $G_n \in \mathbb{G}$ is the label-irrelevant substructure, which is independent to Y. Denote G_{sub} as the arbitrary subgraph. Suppose G_n influences G_{sub} only through G, the following inequality holds:

$$I(G_{sub}, G_n) \leq I(G_{sub}, G) - I(G_{sub}, Y), \qquad (7)$$

Lemma 4.1 indicates when setting $\beta = 1$ in Eq. 1, the GIB objective upper bounds the mutual information of G_{sub} and G_n. That is to say, optimizing the GIB objective encourages G_{sub} to be less related to the label-irrelevant substructure G_n. $I(G_n, G_{sub})$ is minimal when G_{sub} is the IB-subgraph. The proof of Lemma 4.1 is in the Supplementary Materials. We next give the theorem that minimization of VGIB objective in Eq. 6 also leads to the irrelevance of G_{sub} and G_n.

Theorem 4.1. *Let $G \in \mathbb{G}$ and $Y \in \mathbb{R}$ be the graph and its label. $G_n \in \mathbb{G}$ is the label-irrelevant substructure in G,*

which is independent to Y. Denote $G_\epsilon \in \mathbb{G}$ as the subgraph formed by injected noise. Then, if we choose the subgraph G_{sub} by dropping G_ϵ in G_N, the following inequality holds:

$$I(G_{sub}, G_n) \leq I(G_N, G_n) \leq I(G_N, G) - I(G_N, Y) \qquad (8)$$

Please refer to the Supplementary Materials for the proof of Theorem 4.1. VGIB differs from GIB mainly in noise injection. In the next section, we will show that this process leads to convenience in optimization.

4.3. Optimization of Variational Graph Information Bottleneck

We first examine the first term $I(G_N, Y)$ in Eq. 6, which encourages G_N is informative of graph label Y.

$$-I(G_N, Y) \leq \mathrm{E}_{Y, G_N} - \log q_\theta(Y|G_N)$$
$$:= \mathcal{L}_{cls}(G_N, Y), \qquad (9)$$

where $q_\theta(Y|G_N)$ is the variational approximation to $p(Y|G_N)$. $q_\theta(Y|G_N)$ outputs the label distribution of G_N and can be modeled as a classifier. $\mathcal{L}_{cls}$ is the classification loss. We choose the cross-entropy loss and mean square loss for categorical Y and continuous Y, respectively.

For the second term $I(G_N, G)$ in Eq. 6, we first obtain the graph representation z_N of G_N via a readout function. We employ the sufficient encoder assumption [42] that the information of z_N is lossless in the encoding process, leading to $I(z_N, G) \approx I(G_N, G)$. By choosing the distribution of noise and the readout function as the Gaussian distribution, $I(G_N, G)$ has a tractable variational upper bound.

Proposition 4.1 (Variational upper bound of $I(G_N, G)$)**.** Let m_G be the number of nodes in G. h_j is the j-th node representation of G. $\epsilon_G \sim \mathcal{N}(\mu_h, \sigma_h^2)$ is the noise sampled from the Gaussian distribution. μ_h, σ_h^2 are mean and variance of h_j in G. Suppose the readout function is chosen

	Input	GCN+VGIB	GCN+GIB	GCN+Att05	GCN+Att07
QED					
Value	0.86	0.90	0.64	Invalid	0.56
HLM-CLint					
Value	2.01	2.15	1.62	1.05	1.83
MLM-CLint					
Value	2.39	2.26	1.32	0.31	-0.75
RLM-CLint					
Value	2.31	2.26	2.26	0.74	1.70

Figure 3. Qualitative results on graph interpretation. VGIB can generate more precise interpretations of the input molecules.

Table 1. Mean and standard deviation of absolute property divergence between the input molecules and the found subgraphs. The results of the baselines are copied from the existing literature [52]. The lower the better.

Method	QED	HLM-CLint	MLM-CLint	RLM-CLint
GCN+Att05	0.48±0.07	0.90±0.89	0.92±0.61	1.17±0.63
GCN+Att07	0.41±0.07	1.18±0.60	1.69±0.88	1.22±0.85
GCN+GIB	0.38±0.12	0.37±0.30	0.72±0.55	1.15±0.68
GCN+VGIB	**0.32±0.12**	**0.34±0.28**	**0.69±0.58**	1.02±0.64

Table 2. Training time of different methods on QED dataset.

Method	GCN+Att05	GCN+Att07	GCN+GIB	GCN+VGIB
Time	142.01s	141.18s	1712.93s	146.53s

5. Experiments

We extensively evaluate the proposed method on three tasks, i.e., graph interpretation, post-hoc explanation of GNN and graph classification. For the first task, we aim to verify whether VGIB can recognize the substructures that interpret the properties of the input molecules [52] or not. Secondly, we employ VGIB to generate post-hoc explanation of the GNNs. For the third task, we plug VGIB into various GNN baselines to see whether the found IB-subgraph can boost the performance of graph classification.

5.1. Graph Interpretation

In this experiment, we extract the substructures which have the most similar properties to the original molecules. We consider four properties: QED, HLM-CLint, MLM-CLint, and RLM-CLint. QED measures the probability of a molecule being a drug within the range of $[0, 1.0]$. HLM-CLint, MLM-CLint, and RLM-CLint are estimated values of in vitro human, mouse and rat liver microsome metabolic stability, respectively (base 10 logarithm of mL/min/g)[2].

from mean or sum. Then up to a constant, the variational upper bound of $I(G_N, G)$ is:

$$\mathcal{L}_{\text{MI}}(Z_N, G) \leq \mathrm{E}_G(-\frac{1}{2}\log A_G + \frac{1}{2m_G}A_G + \frac{1}{2m_G}B_G^2) \tag{10}$$

where $A_G = \sum_{j=1}^{m_G}(1 - \lambda_j)^2$ and $B_G = \frac{\sum_{j=1}^{m_G}\lambda_j(h_j - \mu_h)}{\sigma_h}$. Please refer to the Supplementary Materials for the proof of Proposition 4.1. One can efficiently estimate Eq. 9 and Eq. 10 with the batched data in the training set. The overall loss is:

$$\mathcal{L} = \mathcal{L}_{\text{cls}}(G_N, Y) + \beta\mathcal{L}_{\text{MI}}(Z_N, G) \tag{11}$$

---

[2]We evaluate QED values of molecules with the toolkit on https://www.rdkit.org/. Moreover, we obtain HLM-CLint, MLM-CLint, and RLM-CLint value of molecules on https://drug.ai.tencent.com/.

Table 3. Performance of different methods on explaining the predictions of GCN in terms of fidelity scores. We set the sparsity score of the explanatory subgraphs as 0.5 for a fair comparison.

Metric	Fidelity+↑				Fidelity-↓			
Property	RLM-Clint	HLM-Clint	QED	DRD2	RLM-Clint	HLM-Clint	QED	DRD2
GNNExplainer	0.694	0.778	0.602	0.74	0.478	0.616	0.498	0.433
PGExplainer	0.632	0.692	0.598	0.686	0.502	0.62	0.56	0.54
GraphMask	0.632	0.706	0.602	0.673	0.516	0.592	0.574	0.4866
IGExplainer	0.684	0.758	0.592	0.693	0.602	0.686	0.584	0.58
GraphGrad-CAM	0.67	0.782	0.586	0.659	0.56	0.668	0.564	0.566
GIB	0.654	0.781	0.601	0.724	0.483	0.643	0.525	0.428
VGIB	**0.765**	**0.792**	**0.627**	**0.756**	**0.463**	**0.579**	**0.487**	**0.424**

We collect molecules with QED$\geq$ 0.85, HLM-CLint$\geq$ 2, MLM-CLint$\geq$ 2, and RLM-CLint$\geq$ 2 from ZINC250K [17], and we individually build datasets with the selected molecules for the four properties. For each property, we use 85%, 5%, and 10% of the molecules for training, validating, and testing, respectively. Please refer to the Supplementary Materials for the statistics of datasets.

We compare the proposed VGIB with GIB [52] and the attention-based method [25]. For VGIB, we first learn the node representation with a GCN and inject noise into each node as shown in Eq. 4 and Eq. 3. Then we obtain the perturbed graph representation by pooling the noisy node representations with a readout function. We thereafter optimize the loss function in Eq. 11 and collect the nodes with $p_i \geq 0.5$ in Eq. 3 to obtain the IB-subgraph. We simultaneously supervise the classifier in Eq. 9 with the representation of the input graph and its label to enhance the informativeness of the subgraph. For GIB, we follow the existing method [52] and optimize the model in a bilevel optimization scheme. As for the attention-based method, we attentively pool the node representations with the attention scores for label prediction. Furthermore, we select the nodes with top 50% and 70% attention scores to form the predictive subgraphs. For each method, we adopt a 2-layer GCN with 16 hidden dimensions for a fair comparison. We run experiments on one TITAN RTX GPU. If the found subgraph is disconnected, we choose its largest connected part to ensure chemical validity.

We report the mean and standard deviation of absolute property divergence between the input molecules and the found subgraphs in Table 1. We quantitatively compare the performance of different methods on the interpretation of graph properties. The proposed VGIB method shows favorable performance against GCN+GIB. For the attention-based method, the performance is sensitive to the selected value of the threshold since the results of GCN+Att05 vary from those of GCN+Att07. Therefore, one needs to fine-tune the threshold for different tasks. In contrast, our VGIB is free from a manually selected threshold thanks to the information-theoretic objective.

We then compare the training time of different methods

on the QED dataset. We train different methods 3 times and report the average training time in Fig. 2. It is shown that the attention-based methods achieve the fastest training due to their simple architectures. Although VGIB is slower than the attention-based methods, it achieves significant performance gain in finding subgraphs with more similar properties to the input molecules. Compared with GIB, VGIB trains over ten times faster than GIB with better performance. The reason is that estimating the mutual information in GIB is time-consuming. However, VGIB a enjoys tractable upper bound of its objective, which is easy to optimize.

5.2. Explainability of GCN

As GCN model becomes prevalent in various tasks [14, 54], there is increasing concerns on its explainability since GCN are treated as black-box [49]. In this section, we employ the proposed VGIB to explain the prediction of GCN for molecule classification on ZINC250K dataset. We consider four properties: QED, HLM-CLint, RLM-CLint and DRD2 construct the dataset for each property. For QED, we label the molecules with QED$\geq$0.85 /$<$0.85 as 1/0. For DRD2, we label the molecules with DRD2$\geq$0.50 /$<$0.50 as 1/0. For HLM-CLint and RLM-CLint, we label the molecules with the property value greater than 2.0 as 1 or 0 otherwise. For each property, we train the GCN on the training set and employ VGIB to generate post-hot explanation on the test set. Please refer to Supplementary Materials for more details on the dataset splits and the training process.

We compare VGIB with various explanation model including GNNExplainer [49], PGExplainer [28], GraphMask [37], IGExplainer [41], GraphGrad-CAM [31] and GIB [52]. These methods interpret the prediction of GCN with the node importance score ranging in $[0, 1]$, where 1 indicates the node is the most important to the molecule classification and 0 otherwise. Please refer to Supplementary Materials for more details on the baseline methods.

We employ the fidelity score to evaluate how the explanation is faithful to the GCN model [55]. Specifically, let y_i and $\hat{y}_i$ be the ground-truth and the prediction of the i-th

Table 4. Classification accuracy of graphs with **sum pooling** and **mean pooling** as the readout function. We report the mean and standard deviation of the testing accuracy in 10-fold cross-validation for each method. The bold results are the overall best performances and the underlining results are the best performances of certain backbones.

	Method	MUTAG	PROTEINS	IMDB-B	DD	COLLAB	REDDIT-B
Sum pooling	GCN	0.76±0.09	0.72±0.05	0.71±0.04	0.74±0.03	0.78±0.03	0.75±0.05
	GIB+GCN	0.77±0.07	0.74±0.04	0.72±0.04	0.75±0.05	0.78±0.02	0.77±0.04
	VGIB+GCN	0.79±0.09	0.74±0.04	0.74±0.04	0.77±0.09	**0.80±0.02**	0.82±0.02
	GraphSAGE	0.74±0.08	0.73±0.05	0.71±0.05	0.75±0.03	0.78±0.02	0.78±0.10
	GIB+GraphSAGE	0.75 ±0.07	0.73±0.04	0.72±0.05	0.76±0.04	0.79±0.02	0.80±0.03
	VGIB+GraphSAGE	0.77±0.07	0.74±0.04	0.73±0.03	**0.78±0.05**	**0.80±0.03**	0.80±0.09
	GIN	0.83±0.07	0.74±0.05	0.71±0.05	0.71±0.03	0.78±0.02	0.81±0.10
	GIB+GIN	0.84±0.06	0.74±0.05	0.74±0.07	0.74±0.04	0.78±0.03	0.84±0.03
	VGIB+GIN	**0.86±0.08**	**0.75±0.04**	**0.74±0.05**	0.75±0.04	0.79±0.03	**0.86±0.03**
	GAT	0.75±0.08	0.72±0.04	0.72±0.03	0.75±0.04	0.76±0.04	0.71±0.08
	GIB+GAT	0.75±0.09	0.73±0.04	0.72±0.05	0.76±0.04	0.77±0.03	0.73±0.04
	VGIB+GAT	0.76±0.07	0.73±0.04	0.73±0.07	0.77±0.04	0.79±0.03	0.76±0.05
Mean pooling	GCN	0.72±0.11	0.71±0.04	0.71±0.04	0.72±0.05	0.77±0.02	0.75±0.06
	GIB+GCN	0.74±0.08	0.72±0.05	0.72±0.03	0.74±0.05	0.78±0.02	0.76±0.04
	VGIB+GCN	0.76±0.10	0.73±0.04	0.73±0.04	0.75± 0.10	**0.79±0.02**	0.78± 0.02
	GraphSAGE	0.73±0.07	0.72±0.04	0.70±0.04	0.73±0.04	0.77±0.02	0.78±0.03
	GIB+GraphSAGE	0.74±0.07	0.72±0.04	0.72±0.05	0.74±0.04	0.78±0.03	0.79±0.03
	VGIB+GraphSAGE	0.75±0.08	0.73±0.04	0.73±0.03	**0.76±0.05**	0.78±0.03	0.80±0.03
	GIN	0.82±0.07	0.71±0.06	0.72±0.05	0.73±0.03	0.78±0.03	0.82±0.02
	GIB+GIN	0.83±0.06	0.71±0.05	0.73±0.07	0.74±0.04	0.78±0.03	0.84±0.04
	VGIB+GIN	**0.84±0.09**	0.72±0.05	**0.74±0.05**	0.75±0.04	0.79±0.03	**0.85±0.03**
	GAT	0.74±0.08	0.71±0.04	0.71±0.04	0.70±0.06	0.76±0.03	0.77±0.04
	GIB+GAT	0.75±0.10	0.73±0.04	0.72±0.05	0.72±0.04	0.77±0.03	0.79±0.04
	VGIB+GAT	0.76±0.07	**0.74±0.03**	0.73±0.07	0.73±0.05	0.78±0.03	0.79±0.05

input molecule. Define k as the sparsity score of the explanatory subgraph. The explanatory subgraph is obtained by choosing the nodes with top $k\%$ scores from the input molecule, and we denote its prediction as $\hat{y}_i^k$. The Fidelity- score is computed as follows:

$$Fidelity- = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(y_i = \hat{y}_i) - \mathbb{1}(y_i - \hat{y}_i^k) \quad (12)$$

where $\mathbb{1}(y_i = \hat{y}_i)$ is the indicator function which outputs 1 if $y_i = \hat{y}_i$ and 0 otherwise. Fidelity- score measures how the prediction of the explanatory subgraph is close to the input molecule. The lower value of Fidelity- score indicates more faithful explanation. Similarly, we define $\hat{y}_i^{1-k}$ as the prediction of the complementary subgraph, which is obtained by removing the explanatory subgraph from the input. The Fidelity+ score is defined as follows:

$$Fidelity+ = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(y_i = \hat{y}_i) - \mathbb{1}(y_i - \hat{y}_i^{1-k}) \quad (13)$$

The Fidelity+ score indicates more important nodes are identified in the explanation.

Table 3 shows the fidelity scores of the explanations produced by different methods. The sparsity score is set to be $k = 0.5$ for all the explanatory subgraphs. As shown in table 3, VGIB achieves best fidelity scores on all the properties. This shows that VGIB generate faithful explanation to the predictions of the GCN. Moreover, to comprehensively evaluate different methods, we set $k \in \{0.30, 0.35, 0.40, 0.45, 0.50, 0.55, 0.60\}$ and compare the performance under different sparsity scores. As shown in Figure 4, VGIB generate most faithful explanations to the predictions of GCN in terms of fidelity scores by under different sparsity scores. This shows that VGIB can identify most important nodes to the predictions of GCN.

5.3. Graph Classification

In this subsection, we aim to find out whether the found subgraph can improve the performance of baselines on graph classification or not. We evaluate different methods
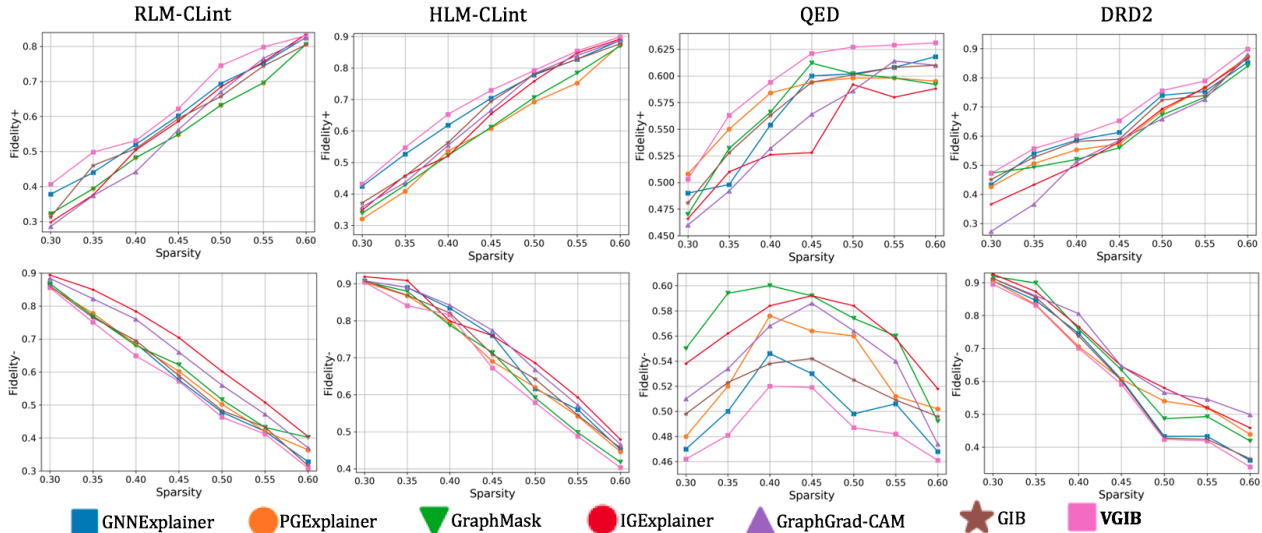
Figure 4. Performance of different methods on explaining the predictions of GCN in terms of fidelity scores ($Fidelity+ \uparrow$ and $Fidelity- \downarrow$). VGIB generates the most faithful explanations to the predictions of GCN by identifying the most important nodes under different sparsity scores.

on MUTAG [36], PROTEINS [6], DD, IMDB-BINARY, REDDIT-BINARY, and COLLAB [35] datasets, which are wildly used for graph classification. Please refer to Supplementary Materials for the statistics of datasets.

We consider four GNN baselines including GCN [22], GAT [44], GraphSAGE [14] and GIN [47]. We use the mean and sum pooling as a readout function for the baseline methods to obtain the graph representation for prediction. Then, similar to GIB [52], we plug VGIB into these baselines. Specifically, we adopt the baseline models to extract node representation. Then, we recognize the IB-subgraph by optimizing the VGIB objective. We pool the node representations in the IB-subgraph for classification with the same readout function as the baselines. For a fair comparison, we adopt a 2-layer network architecture and 16 hidden dimensions for different methods. We train these methods for 100 epochs and test the models with the smallest validation loss. We report the mean and standard deviation of accuracy across 10 folds [3].

The experimental results are summarized in Table 4. Compared with the baselines, VGIB can discover an informative yet compressed subgraph of the input. Therefore, it relieves the perturbation of noise structures and redundant information and boosts the performance of the baselines. VGIB also outperforms the GIB-based methods on most of the datasets.

6. Discussions

Potential Negative Impacts: Our method aims to discover a predictive yet compressed subgraph of the input

graph, and can be deployed in social network analysis and biochemistry. The concern is that if not adequately used under administration, our method potentially leads to the leakage in privacy and intellectual property.

Limitations: We assume the encoding process from G_N to z_N is lossless following the sufficient encoder assumption [42]. Hence we approximate the compression term $I(G_N, G)$ with $I(z_N, G)$ and obtain a tractable variational upper bound for optimization. In fact, the encoding process is not lossless due to the data processing inequality. Thus, we actually approach $I(G_N, G)$ with $I(z_N, G)$ in practice. We leave the in-depth analysis in our future work.

7. Conclusion

We propose a novel Variational Graph Information Bottleneck framework for improved and efficient subgraph recognition. The proposed noise injection method serves as an alternative to compress the information in the discovered subgraph and allows a tractable objective of VGIB for efficient and stable training. Using the proposed method, we make the practical training more than 10 times faster than the existing methods. The experimental results show that the proposed method performs favorably against the existing methods on various tasks.

Acknowledgement

---

[3]We follow the protocol in https://github.com/rusty1s/pytorch_geometric/tree/master/benchmark/kernel

References

[1] Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2897–2905, 2018. 2

[2] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016. 2

[3] Emily Alsentzer, Samuel G Finlayson, Michelle M Li, and Marinka Zitnik. Subgraph neural networks. *arXiv preprint arXiv:2006.10538*, 2020. 1

[4] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, R. Devon Hjelm, and Aaron C. Courville. Mutual information neural estimation. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 530–539, 2018. 3

[5] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 2

[6] Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1):i47–i56, 2005. 8

[7] Jintai Chen, Biwen Lei, Qingyu Song, Haochao Ying, Danny Z Chen, and Jian Wu. A hierarchical graph network for 3d object detection on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 392–401, 2020. 1

[8] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018. 2

[9] Yixiang Fang, Kaiqiang Yu, Reynold Cheng, Laks V. S. Lakshmanan, and Xuemin Lin. Efficient algorithms for densest subgraph discovery. *Proceedings of VLDB Endowment*, 12(11):1719–1732, 2019. 2

[10] Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. *arXiv preprint arXiv:1705.07832*, 2017. 3

[11] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. *Proceedings of the 34th International Conference on Machine Learning*, 70:1263–1272, 2017. 1

[12] Aristides Gionis and Charalampos E. Tsourakakis. Dense subgraph discovery: Kdd 2015 tutorial. In *Knowledge Discovery and Data Mining*, pages 2313–2314. ACM, 2015. 2

[13] Anirudh Goyal, Riashat Islam, Daniel Strouse, Zafarali Ahmed, Matthew Botvinick, Hugo Larochelle, Yoshua Bengio, and Sergey Levine. Infobot: Transfer and exploration via the information bottleneck. In *The International Conference on Representation Learning*, 2019. 2

[14] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017. 1, 2, 6, 8

[15] Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. Adaptive sampling towards fast graph representation learning. *arXiv preprint arXiv:1809.05343*, 2018. 2

[16] Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. In *Advances in neural information processing systems*, 2019. 2

[17] John J Irwin and Brian K Shoichet. Zinc- a free database of commercially available compounds for virtual screening. *Journal of chemical information and modeling*, 45(1):177–182, 2005. 6

[18] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 3

[19] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, pages 2323–2332. PMLR, 2018. 1

[20] Wengong Jin et al. Multi-objective molecule generation using interpretable substructures. In *ICML*, 2020. 1

[21] Jaekyeom Kim, Minjung Kim, Dongyeon Woo, and Gunhee Kim. Drop-bottleneck: Learning discrete compressed representation for noise-robust exploration. *arXiv preprint arXiv:2103.12300*, 2021. 2

[22] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *The International Conference on Representation Learning*, 2017. 1, 2, 8

[23] Huan Lei, Naveed Akhtar, and Ajmal Mian. Seggcn: Efficient 3d point cloud segmentation with fuzzy spherical kernel. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11611–11620, 2020. 1

[24] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9267–9276, 2019. 1

[25] Jia Li, Yu Rong, Hong Cheng, Helen Meng, Wenbing Huang, and Junzhou Huang. Semi-supervised graph classification: A hierarchical graph perspective. In *The World Wide Wed Conference*, 2019. 2, 6

[26] Maosen Li, Siheng Chen, Ya Zhang, and Ivor W Tsang. Graph cross networks with vertex infomax pooling. *arXiv preprint arXiv:2010.01804*, 2020. 1

[27] Zhi-Hao Lin, Sheng-Yu Huang, and Yu-Chiang Frank Wang. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1800–1809, 2020. 1

[28] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *NeurIPS*, 2020. 1, 6

[29] Yawei Luo, Ping Liu, Tao Guan, Junqing Yu, and Yi Yang. Significance-aware information bottleneck for domain adaptive semantic segmentation. In *ICCV*, pages 6777–6786. IEEE, 2019. 2

[30] XueBin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, and Sergey Levine. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. In *The International Conference on Representation Learning*, 2019. 2

[31] Phillip E Pope, Soheil Kolouri, Mohammad Rostami, Charles E Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10772–10781, 2019. 6

[32] Kaizhi Qian, Yang Zhang, Shiyu Chang, David Cox, and Mark Hasegawa-Johnson. Unsupervised speech decomposition via triple information bottleneck. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 2

[33] Ekagra Ranjan, Soumya Sanyal, and Partha Pratim Talukdar. Asap: Adaptive structure aware pooling for learning hierarchical graph representations. In *AAAI*, 2020. 2

[34] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 33, 2020. 1

[35] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015. 8

[36] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole Von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical review letters*, 108(5):058301, 2012. 8

[37] Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. Interpreting graph neural networks for nlp with differentiable edge masking. *International Conference on Learning Representation*, 2020. 6

[38] Karl Schulz et al. Restricting the flow: Information bottlenecks for attribution. *ICLR*, 2020. 2

[39] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000*, 2019. 1, 2

[40] Jianxin Sun, Qi Li, Weining Wang, Jian Zhao, and Zhenan Sun. Multi-caption text-to-face synthesis: Dataset and algorithm. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 2290–2298, 2021. 2

[41] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017. 6

[42] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning. *arXiv preprint arXiv:2005.10243*, 2020. 4, 8

[43] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000. 2, 3

[44] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lia, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representation*, 2017. 1, 2, 8

[45] Rundong Wang, Xu He, Runsheng Yu, Wei Qiu, Bo An, and Zinovi Rabinovich. Learning efficient multi-agent communication: An information bottleneck approach. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 2

[46] Tailin Wu, Hongyu Ren, Pan Li, and Jure Leskovec. Graph information bottleneck. *NeurIPS*, 2020. 1, 2

[47] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks? In *Proceedings of the 7th International Conference on Learning Representations*, ICLR '19, pages 1–17, 2019. 2, 8

[48] Xifeng Yan and Jiawei Yan. gspan: graph-based substructure pattern mining. In *IEEE International Conference on Data Mining*, pages 721–724, 2002. 2

[49] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in neural information processing systems*, 2019. 1, 2, 6

[50] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Advances in neural information processing systems*, 2018. 2

[51] Junchi Yu, Jie Cao, Yi Li, Xiaofei Jia, and Ran He. Pose-preserving cross spectral face hallucination. In *IJCAI*, pages 1018–1024, 2019. 2

[52] Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Graph information bottleneck for subgraph recognition. *International Conference on Learning Representation*, 2021. 1, 2, 3, 5, 6, 8

[53] Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Recognizing predictive substructures with subgraph information bottleneck. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 1

[54] Junchi Yu, Tingyang Xu, Yu Rong, Junzhou Huang, and Ran He. Structure-aware conditional variational auto-encoder for constrained molecule optimization. *Pattern Recognition*, 126:108581, 2022. 6

[55] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. *arXiv preprint arXiv:2102.05152*, 2021. 2, 6

[56] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 2

[57] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pages 11458–11468. PMLR, 2020. 2