Interpretable Deep Graph Generation with Node-edge Co-disentanglement

Xiaojie Guo
xguo7@gmu.edu
George Mason University
Fairfax, VA, USA

Liang Zhao∗
lzha9@gmu.edu
George Mason University
Fairfax, VA, USA

Zhao Qin
zqin02@syr.edu
Syracuse University
Syracuse, NY, USA

Xiaojie Guo, Liang Zhao, Zhao Qin, Lingfei Wu, Amarda Shehu, and Yanfang Ye. 2020. Interpretable Deep Graph Generation with Node-edge Co-disentanglement. In Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’20), August 23–27, 2020, Virtual Event, CA, USA. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3394486.3403221

1 INTRODUCTION

Recent advances in deep generative models, such as variational auto-encoders (VAE) [24] and generative adversarial networks (GAN) [16], have made important progress towards generative modeling for complex domains, such as image data. The goal here is to learn the underlying (low-dimensional) distribution of the images, hence image generation is treated as sampling from learned distributions. Building on these techniques for images, which can be considered as grid-structured data, a special case of graphs, a number of deep learning models for generating general graphs have been proposed over the last couple of years [25, 32, 43]. These involves real-world applications such as modeling physical and social interactions [9, 28], discovering new chemical and molecular structures, and constructing knowledge graphs.

When we learn the underlying distribution of complex data such as images, learning interpretable representations of data that expose semantic meaning is very important. Such representations are useful not only for standard downstream tasks such as supervised learning and reinforcement learning, but also for tasks such as transfer learning and zero-shot learning where humans excel but machines struggle [29]. As yet, most research has focused on learning factors of variations in the data, commonly referred to as learning a disentangled representation, where the variables of the representation are highly independent. Examples of this include variables that only control the size of objects, or their color. For the instance in Fig. 1 (a), where a semantic factor controls the degree of smile in a human facial image.

However, in the promising domain of deep generative models for graph generation, disentangled enhancement has rarely been well explored yet, but could be highly beneficial for applications such as controlling the generation of protein structures, or designing...
Internet of Things (IoT).

As shown in Fig. 1, we would love to generalize from an image situation to a graph situation, where the variables control specific factors related to node attributes, edge attributes, or joint-node-edge patterns in the graph. For example, Fig. 1(a) shows the semantic factor (i.e., smile) in the images, which can be regarded as special cases of graphs where nodes are pixels that are connected in a fixed topology. All the factors that control image formulation are effectively node-related. Fig. 1(b) shows the factors that control the formulation of a cyber network, which is an attributed graph where computers are nodes and their links are edges. Unlike images, there are three types of factors formulating the networks: (1) node-related factors that control some properties of node attributes but are independent of edge patterns (e.g., the CPU usage of each computer); (2) edge-related factors that only influence edge patterns but are independent of node patterns (e.g., geo-spatial distances between computers); and (3) node-edge-joint factors that jointly influence some properties from both nodes and edges (e.g., the node patterns of "downloaded data amount" and edge patterns "network traffic" which are inherently highly entangled and hence must be controlled by such factors). Thus, it is necessary to develop a generic model to discover and disentangle all three types of factors for the graph data. Though a few researchers have sought to apply the disentanglement learning to graphs [33, 36, 44], as yet they only have identified the latent factors that caused the edge between a node and its neighbors.

Node-related Factor (smile)

(a) Semantic factors for images

Node-related Factor (e.g., CPU usage):

Edge-related Factor (e.g., geo-spatial distance among computers):

Node-edge-joint Factor (e.g., data downloading amount/network traffic):

(b) Semantic factors for graphs

Figure 1: Two examples of disentanglement: (a) semantic factors of images, where each pixel is a node and each pixel is connected to its eight neighboring pixels, and (b) semantic factors of cyber networks, where each computer is a node and the link between each pair of computers is an edge (better seen in color).

In this paper, we focus on the generic problem of disentanglement learning on attributed graphs, where the characteristics of graphs pose great challenges to disentanglement learning on graphs: 1) \textbf{Lack of node and edge joint deconvolution operations.} The formation process for real-world graphs, which is both complex and iterative, is based on the three types of factors depicted in Fig. 1. For example, edges are generated not only by the edge related factors, but also node-edge-joint related factors. There is no existing graph decoder that can simultaneously handle all three types of factors during the generation process. 2) \textbf{Complex disentanglement enhancement of multiple types of latent representation.} Although the three types of semantic factors mentioned in Fig. 1 are independent from each other, it is extremely difficult to enforce that. First, it is difficult to automatically categorize individual factors into these three types. Second, even they are categorized, the enhancement of such independency patterns still cannot be accomplished by the existing techniques which mostly focus on images without categorization capability. 3) \textbf{The dilemma between disentanglement and reconstruction quality for attributed graphs.} Disentangling the three types of factors and reconstructing both edges and nodes can require multiple trade-offs between reconstruction errors and disentanglement performance during the training process. For example, the objective of disentanglement of node-edge-joint factors can conflict with not only edge but also node reconstruction errors. Existing methods cannot handle the situation of attributed graphs.

To the best of our knowledge, this is the first work that can address all the above challenges and provides a generic framework that incorporates multiple disentanglement enhancements for attributed graphs. We propose the new Node-Edge Disentangled Variational Auto-encoder (NED-VAE) model, a deep unsupervised generative approach for disentanglement learning on graphs that automatically discovers the independent latent factors in both edges and nodes. A novel objective for node-edge jointly disentanglement is derived and proposed based on the variational autoencoder (VAE) [24, 38]. A novel architecture is proposed consisting of three sub-encoders and two sub-decoders to model the complicated relationships between nodes and edges. We also propose a general framework of objectives that can include various extensions of the base NED-VAE to realize the group-wise and variable-wise disentanglement. The contributions of this work are summarized as follows:

- \textbf{A novel framework is proposed for the disentanglement of attributed graph generation.} In order to jointly disentangle the nodes and edges, we derive a novel objective framework for learning three factors that are exclusive to node patterns, exclusive to edge patterns, and those spanning node-edge-joint patterns. This new framework is demonstrated to be a significant generalization over existing disentanglement frameworks for image generation.

- \textbf{A novel architecture is proposed for disentanglement learning on graphs.} Derived from the theoretical objective of our framework, a novel architecture proposed for the representation learning of graphs consists of three sub-encoders (a node encoder, an edge encoder, and a node-edge co-encoder) to learn the three types of representations, along with two novel sub-decoders (a node-decoder and an edge decoder) to co-generate both nodes and edges.

- \textbf{Simultaneous group-wise and variable-wise disentanglement.} The proposed framework hierarchically disentangles attributed graph generation according to node, edge, and...
their joint factors. A set of variational auto-encoder-based models for attributed graphs have been proposed.

- Comprehensive experiments have been conducted to validate the effectiveness of our proposed model and its extensions. Qualitative and quantitative experiments on two synthetic and two real-world datasets demonstrate that NED-VAE and its extensive models are indeed capable of learning disentangled factors for different types of graphs.

2 RELATED WORKS

Disentanglement Learning. Disentangled representation learning has gained considerable attention, in particular in the field of image representation learning [1, 6, 22, 23]. The goal is to learn representations that separate out the underlying explanatory factors responsible for variations in the data. Such representations have been shown to be relatively resilient to the complex variants involved [3], and can be used to enhance generalizability as well as improve robustness against adversarial attack [1]. The disentangled representations are inherently more interpretable, and can thus potentially facilitate debugging and auditing [11]. This has prompted a number of approaches that modify the VAE objective by adding, removing, or altering the weight of individual terms [1, 6, 14, 23, 27, 35, 49]. However, the best way of learning representations that disentangle the latent factors behind a graph remains largely unexplored.

Graph neural networks. Recent work on graph neural networks (GNNs) [17, 41], especially graph convolutional networks [5, 21], is attracting considerable attention, because of their remarkable success in multiple domains such as natural language processing [7, 8], computer vision [42], software engineering [30] and traffic flow prediction [31]. Graph Convolutional Networks originated from spectral graph convolutional neural networks [5], which were then extended by using fast localized convolutions [10], and further approximated by an efficient architecture for a semi-supervised setting proposed by Kipf and Welling [25]. Self-attention mechanisms and sub-graph-level information have also been explored as ways to potentially improve the representation power of learned node embeddings [2, 15, 45].

Graph generation. Most of the existing GNN based graph generation methods are based on VAE [40, 43] and generative adversarial nets (GANs) [4], and others [32, 48]. For example, GraphRNN [48] builds an autoregressive generative model on these sequences utilizing LSTM model and has demonstrated good scalability; while GraphVAE [43] represents each graph in terms of its adjacent matrix and feature vector and utilizes the VAE model to learn the distribution of the graphs conditioned on a latent representation at the graph level. Graphite [18] and VEGA [26] encode the nodes of each graph into node-level embeddings and predict the links between each pair of nodes to generate a graph. Some conditional graph generation methods also provide powerful graph encoders and decoders for attributed graphs where both node and edge attributes are considered [19, 20].

3 PROBLEM FORMULATION

Define an input graph as \( G(V, E, F) \), where \( V \) is the set of \( N \) nodes and \( E \subseteq V \times V \) is the set of \( M \) edges. \( E \) contains all pairs of nodes, while the existence of each edge is reflected by one of its attributes. \( E \in \mathbb{R}^{N \times N \times K} \) is the edge attributes tensor, where \( K \) is the dimension of the edge attributes. \( F \in \mathbb{R}^{N \times D} \) refers to the node attribute matrix, where \( F_i \in \mathbb{R}^{1 \times D} \) is the node attributes of node \( i \) and \( D \) is the dimension of the node attribute vector. As shown in Fig. 1, three types of factors (i.e. node-related factors, edge-related factors and node-edge joint related factors) are assumed to control the generation of the graph \( G \).

The goal is to develop an unsupervised deep generative model that can learn the joint distribution of the graph \( G \) and three groups of generative latent variables \( Z = (z_e \in \mathbb{R}^{L_1}, z_f \in \mathbb{R}^{L_2}, z_g \in \mathbb{R}^{L_3}) \) \((L_1, L_2, \text{and} L_3 \text{are the number of variables in each group})\) to discover the three types of factors, such that the observed graph \( G \) can be generated as \( p(G | Z) = P(E, F | z_e, z_f, z_g) \). There are three challenges must be overcome to achieve the above goal: (1) The lack of co-decoder based on co-deconvolution for the generation of attributed graph \( G \) that is capable of jointly generating both the nodes attributes \( F \) and edges attributes \( E; \) (2) difficulty of enforcing independence among the variable groups \( z_e, z_f \) and \( z_g \) (group-wise disentanglement), rather than simply enforcing the disentanglement of the variables inside \( z_e, z_f \) and \( z_g \) (variable-wise disentanglement); and (3) the need to simultaneously solve multiple reconstruction-disentanglement conflicts in \( z_e \) and \( E, z_f \) and \( F, z_g \) and \( E \) and \( z_g \) and \( F \).

4 NODE-EDGE DISENTANGLEMENT VAE

In this section, we first introduce the derived training objective and the architecture of the proposed Node-edge Disentanglement VAE (NED-VAE). Then we propose a generic objective framework as well as its derivation to further enforce the disentanglement of NED-VAE models with different purposes. At last, the time and memory complexities of the proposed NED-VAE are analyzed and compared with the existing methods.

4.1 Objective and Architecture

In this section, we first derive the objective for learning disentanglement on graphs. Then, to solve the first challenge, we propose a new architecture, the NED-VAE, based on the derived objectives. NED-VAE includes a novel co-decoder-based co-decoder that is capable of jointly generating nodes and edges.

4.1.1 The objective for disentanglement on graphs. Inspired by the disentanglement learning in the image domain, a suitable objective is to maximize the marginal (log-)likelihood of the observed graph \( G \) in expectation over the whole distribution of latent factors set \( Z \):

\[
\max_{\theta} \mathbb{E}_{p(Z)} [p(E, F | z_e, z_f, z_g)].
\]

For a given observation \( G = (E, F) \), we describe the inferred posterior configurations of the latent factors \( Z = (z_e, z_f, z_g) \) using a probability distribution \( q_{\phi}(z_e, z_f, z_g | E, F) \). Our aim is to ensure that the inferred latent factors \( q_{\phi}(z_e, z_f, z_g | E, F) \) capture all three types of generative factors in a disentangled manner. In order to encourage this disentangling property in the inferred \( q_{\phi}(z_e, z_f, z_g | E, F) \), we can introduce a constraint by trying to match it to a prior \( p(z_e) \), \( p(z_f) \) and \( p(z_g) \) that both controls the capacity of the latent information bottleneck, and embodies the statistical independence.
mentioned above. This can be achieved if we set the prior to be an isotropic unit Gaussian, i.e. \( p(Z) = p(z_e, z_f, z_g) = N(0, I) \), leading to the constrained optimisation problem in Eq. 2, where \( \epsilon \) specifies the strength of the applied constraint:

\[
\max_{\beta, \phi} \mathbb{E}_{G \sim D} \left[ \mathbb{E}_{q_\phi(Z|G)} [\log p_\theta(E,F | z_e, z_f, z_g)] \right] \\
\text{s.t.} \quad D_{KL}(q_\phi(z_e, z_f, z_g | E, F) || p(z_e, z_f, z_g)) < \epsilon. \tag{2}
\]

Eq. 2 can be rewritten as a Lagrangian under the KKT conditions and, according to the complementary slackness KKT condition, we therefore arrive at the \( \beta \)-VAE [22] formulation, which takes the form of the familiar variational free energy objective function:

\[
\mathcal{L}(\theta, \phi, G, Z, \beta) = \mathbb{E}_{q_\phi(Z|G)} [\log p_\theta(E,F | z_e, z_f, z_g)] \\
- \beta D_{KL}(q_\phi(z_e, z_f, z_g | E, F) || p(z_e, z_f, z_g)). \tag{3}
\]

Based on the definitions of \( z_f, z_e, \) and \( z_g \), namely that \( z_f \) only controls some properties of nodes, \( z_e \) only controls some properties of edges and \( z_g \) controls the properties of both, we obtain:

\[
q_\phi(z_e, z_f, z_g | E, F) = q_\phi(z_f | F)q_\phi(z_e | E)q_\phi(z_g | E, F) \tag{4}
\]

\[
p_\theta(E,F | z_e, z_f, z_g) = p_\theta(F | z_f, z_g)p_\theta(E | z_e, z_g) \tag{5}
\]

We can now rewrite the loss function as:

\[
\mathcal{L}(\theta, \phi, G, Z, \beta) = \mathbb{E}_{q_\phi(Z|G)} [\log p_\theta(F,E | z_e, z_f, z_g)] \\
- \beta D_{KL}(q_\phi(z_f | F)q_\phi(z_e | E)q_\phi(z_g | E, F) || p(z_f)p(z_g)) \\
- \mathbb{E}_{q_\phi(Z|G)} [\log p_\theta(E | z_e, z_f, z_g)] \\
- \beta D_{KL}(q_\phi(z_f | F) || p(z_f)) \tag{6}
\]

Given that the goal is to maximize the above objective, a deep generative model is needed to model each of the components in this objective.

4.1.2 The architecture of the node-edge disentangled VAE. Based on the above inference for the objective, we are proposing the Node-Edge Disentangled VAE model (NED-VAE) based on a novel architecture. The architecture of the proposed model is shown in Fig. 2.

The overall framework is based on the traditional VAE, where the encoder learns the mean and standard deviation of the latent representation of the input and the decoder decodes the sampled latent representation vector to reconstruct the input. Unlike the structure of traditional VAE, the proposed framework has three encoders, each of which models the distributions \( q_\phi(z_f | F) \), \( q_\phi(z_g | E, F) \) or \( q_\phi(z_e | E) \), and two novel decoders to model \( p_\theta(F | z_g, z_f) \) and \( p_\theta(E | z_g, z_e) \), that jointly generate the node and edge attributes based on the three types of latent representations. Each type of representations is sampled by their own mean and standard deviation. For example, the representation vectors \( z_f \) are sampled as \( z_e = \mu_f + \sigma_f \otimes \epsilon \), where \( \epsilon \) follows a standard normal distribution. This architecture also partially solves the second challenge described above because it enforces the disentanglement between the two groups of variables \( z_e \) and \( z_f \) by separating their inference process. The details of each components are described as follows.

\[\text{Node, edge and graph encoder.}\] The node encoder consists of several traditional convolution layers to extract latent features from node attribute matrix \( F \); and two paths of fully connected layers to get the mean \( \mu_f \) and standard derivation vectors \( \sigma_f \) of the node representation distribution. The edge encoder consists of several edge convolution layers proposed by Guo et al. [19] to extract edge representations from the edge attribute tensor \( E \); and edge embedding layers to get the node-level representation; and fully connected layers to yield the mean \( \mu_e \) and standard derivation \( \sigma_e \) vectors of the edge representation distribution. The graph encoder consists of several graph convolution layers proposed in [25] to get node-level representations; and fully connected layers to aggregate the learned node representations into a graph-level representation that can be separately mapped into the mean \( \mu_g \) and standard derivation \( \sigma_g \) vectors of the graph representation distribution.

\[\text{Node decoder.}\] The proposed node decoder aims to generate the node attribute matrix \( F' \) based on the sampled node representations \( z_f \) and graph representations \( z_g \), which ensures the node attribute generation process is controlled by both the node-related factors and the node-edge-joint related factors. As shown in Fig. 2 (a), the node decoder consists of several traditional deconvolution layers and fully connected layers as a reversed process of the node encoder.

First, the input node representation \( z_f \) and graph representation \( z_g \) are concatenated together and mapped into several fully connected layers to decode the vector into multiple feature vectors. Next, we aim to convert each feature vector into a feature matrix, where each row should refer to an individual node, by replicating each feature vectors \( N \) times. Moreover, to ensure the diversity and randomness of the nodes in each graph, a node assignment vector \( S \in \mathbb{R}^N \) (shown as a red rectangle in Fig. 2 (a) is sampled following the

---

1 Operation details of the encoders can be found in https://github.com/xguo7/NED-VAE.
normal distribution and is concatenated with each feature matrix. Thirdly, once the feature matrix has been obtained, one-dimensional filters are used to deconvolute each row of the feature matrix into the attribute vectors for each node, completing the reconstruction of the input node attribute matrix \( F \).

**Edge Decoder.** The proposed edge decoder aims to generate the reconstructed edge attribute \( E \) based on the sampled node representations \( z_e \) and graph representations \( z_g \), ensuring that the edge attributes generation is controlled by both the edge-related and node-edge-joint related factors. The proposed edge decoder consists of several edge deconvolution layers and fully connected layers as a reversed process of the edge encoder. The input is the concatenation of both the edge representation \( z_e \) and the graph representation \( z_g \). First, the input vector is mapped into a node-level feature vector through a fully connected layer and is converted into a matrix by being replicated. The same node assignment vector \( S \) is also concatenated to this feature matrix. The hidden feature matrices are then generated by the edge-node deconvolution layer [19] by decoding each of the node-level representations, where the principle is that each node’s representation can make contributions to the generation of its related edges features (contributions are shown as dark grey rectangles in Fig. 2 (b)). Thirdly, the edge-attribute tensor \( E \) is generated through the edge-edge deconvolution layer, where the principle is that each hidden edge feature can contribute to the generation of its adjacent edges.

### 4.2 Framework of node-edge co-disentanglement

To solve the second and third challenges, we propose a generic objective framework to further enforce the disentanglement of NED-VAE models with different purposes. In Section 4.2.1, the basic overall framework with four terms are introduced, namely two conditional distribution terms of the graphs (denoted as (\( \mathbb{A}_E \)), the latent representation term (denoted as (\( \mathbb{B}_f \)), the marginal distribution term for the graphs (denoted as (\( \mathbb{C}_z \)), and the inferred prior distributions (denoted as (\( \mathbb{D}_f \))). In Section 4.2.2, we move on to further enforce the disentanglement among variable groups as explained in the second challenge, generalizing the term (\( \mathbb{C}_z \)) to introduce a novel node-edge-total-correlation term (denoted as (\( \mathbb{A}_f \))) for group-wise disentanglement and a variable-wise disentanglement term (denoted as (\( \mathbb{C}_f \))). Next, in Section 4.2.3, we further enforce the disentanglement inside the three types of latent representations, generalizing the term (\( \mathbb{C}_f \)) to introduce three variable-total-correlation terms (denoted as (\( \mathbb{B}_f \)), (\( \mathbb{A}_f \)), and (\( \mathbb{C}_f \))). Furthermore, based on the framework, six extensions of the base NED-VAE models are proposed that enforce different terms, as shown in Table 1. The existing disentanglement methods in the image domain are proven to be special cases of our generic framework.

**4.2.1 Overall graph disentanglement framework.** As proved by Esmaeili et al. [14], the VAE objective can be equivalently defined as a KL divergence between the generative model \( p_{\theta}(x, z) \) and inference model \( q_{\phi}(x|z)p(z) \). Inspired by this and given that \( p(z_1, z_2, z_3) = p(z_1)p(z_2)p(z_3) \), in conjunction with Eq. 5, the NED-VAE objective for the graph data can be defined as:

\[
\mathcal{L}(\theta, \phi, G, Z, \beta) = -D_{KL}(p_{\theta}(z_e, z_f, z_g|E, F)||q_{\phi}(E, F, z_e, z_f, z_g)) + \mathbb{E}_{q_{\phi}(Z,G)}[log \frac{p_{\theta}(E, F|z_e, z_f, z_g)}{q_{\phi}(E, F, z_e, z_f, z_g)} + log \frac{q_{\phi}(E, F, z_e, z_f, z_g)}{q_{\phi}(E, F, z_e, z_f, z_g)}]
\]

\[
= \mathbb{E}_{q_{\phi}(Z,G)}[log \frac{p_{\theta}(E, F|z_e, z_f, z_g)}{q_{\phi}(E, F, z_e, z_f, z_g)}] - KL(q_{\phi}(E, F)||p_{\theta}(E, F)) - KL(q_{\phi}(z_e, z_f, z_g)||p(z_e, z_f, z_g))
\]

\[
= \mathbb{E}_{q_{\phi}(Z,G)}[log \frac{p_{\theta}(E|z_e, z_f, z_g)}{q_{\phi}(E||z_e, z_f, z_g)}]
\]

\[
- KL(q_{\phi}(E)||p_{\theta}(E)) - KL(q_{\phi}(z_e, z_f, z_g)||p(z_e, z_f, z_g)) \tag{7}
\]

Specifically, Terms \( \mathbb{A}_f \) and \( \mathbb{C}_f \) enforce consistency between the marginal distributions over \( G = (E, F) \) and \( Z = (z_e, z_f, z_g) \). Minimizing the KL divergence in Term \( \mathbb{C}_f \) maximizes the marginal likelihood \( E_{q_{\phi}(E,F)}[logp_{\theta}(E,F)] \); maximizing Term \( \mathbb{A}_f \) which is named as inferred priors term enforces the distance between \( q_{\phi}(z_e, z_f, z_g) \) and \( p(z_e, z_f, z_g) \). Term \( \mathbb{A}_f \) and \( \mathbb{B}_f \) enforce consistency between the conditional distributions. Specifically, Term \( \mathbb{A}_f \) maximizes the correlation for each \( Z \) that generates each \( G \); when \( Z \sim q_{\phi}(Z|G) \) is sampled, the likelihood \( p_{\theta}(G|Z) \) should be higher than the marginal likelihood \( p_{\theta}(G) \). Meanwhile Term \( \mathbb{B}_f \) regularizes Term \( \mathbb{A}_f \) by minimizing the mutual information \( I(Z, G) \) in the inference model.

Since Term \( \mathbb{B}_f \) actually represents the marginal information between the latent \( z_e, z_f, z_g \) and the graphs \( G \), this will lead to poor reconstructions when enforcing disentanglement with high values of \( \beta \) in the proposed NED-VAE [37]. Thus, to solve the trade-off problems between the disentanglement of \( z_e, z_f, z_g \) and \( G \), we propose to either enforce Term \( \mathbb{C}_f \) alone or enforce it with high weights. Accordingly, we can refer to the model enforcing only Term \( \mathbb{C}_f \) as (Node-edge disentangled Inferred Priors VAE) NED-IPVAE-I, and the model enforcing both \( \mathbb{A}_f \) and \( \mathbb{C}_f \) with different weights as NED-IPVAE-II, as shown in Table 1.

**4.2.2 Generalization of the Inferred Priors Term \( \mathbb{A}_f \).** Next, to further address the second challenge and enforce the disentanglement among groups of variables \( z_e, z_f \) and \( z_g \), we further generalize the Term \( \mathbb{A}_f \) by decomposing it and introduce the Node-edge Total Correlation term (\( \mathbb{A}_f \)) in Table 1. Specifically, Term \( \mathbb{A}_f \) can be decomposed into sub components \( \mathbb{A}_e \), \( \mathbb{B}_e \) and \( \mathbb{C}_e \), as the followings.
Table 1: Summary of objectives of the extensions of NED-VAE model. \((\odot)^+\) refers to the sum of \(\odot_e, \odot_f\) and \(\odot_g; \odot^+\) can be changed to \(\odot^+_e, \odot^+_f, \odot^+_g\)

| NED-VAE          | \(\odot^+_1 + \odot^+_2 + \odot^+_3 + \odot^+_4\) |
|------------------|--------------------------------------------------|
| NED-IPVAE-I      | \(\odot^+_1 + \odot^+_2 + \odot^+_3 + \odot^+_4\) |
| NED-IPVAE-II     | \(\odot^+_1 + \odot^+_2 + \odot^+_3 + \odot^+_4\) |
| NED-HCVAE        | \(\odot^+_1 + \odot^+_2 + \odot^+_3 + \odot^+_4\) |
| NED-TVCVAE       | \(\odot^+_1 + \odot^+_2 + \odot^+_3 + \odot^+_4\) |
| NED-AnchorVAE    | \(\odot^+_1 + \odot^+_2 + \odot^+_3 + \odot^+_4\) |

(Here, we use \(Z\) to denote \((z_e, z_f, z_g)\) for clarity):

\[
-D_{KL}(q_\phi(Z)\|p(z_e)p(z_f)p(z_g))
\]

\[
= -E_{q_\phi(Z)}[\log q_\phi(Z) + \log \frac{q_\phi(z_e)q_\phi(z_f)q_\phi(z_g)}{p(z_e)p(z_f)p(z_g)}] 
+ \log \frac{p(z_e)p(z_f)p(z_g)}{p(Z)} 

= -E_{q_\phi(z_e)}[\log \frac{q_\phi(z_e)}{q_\phi(z_e)q_\phi(z_f)q_\phi(z_g)}] 
+ \log \frac{p(z_e)p(z_f)p(z_g)}{p(Z)} 

= -D_{KL}(q_\phi(z_e)\|p(z_e)) - D_{KL}(q_\phi(z_f)\|p(z_f)) - D_{KL}(q_\phi(z_g)\|p(z_g))
\]

We refer to Term \(\mathcal{A}\) as the “Node-Edge Total Correlation” term since it measures the dependence between the three types of latent of graphs \(z_e, z_f\) and \(z_g\) (group-wise disentanglement). The penalty for this term forces the model to find statistically independent factors for the nodes, the edges and their combinations. A heavier penalty on this term induces better separately and disentangled learning for the graph format data. We refer to Term \(\mathcal{C}\) as the “variable-disentanglement” term which enforces the disentanglement of the variables inside each latent group. This allows us to propose variant model which only penalizes Terms \(\mathcal{A}\) and \(\mathcal{C}\), shown as the Node-edge Disentangled Total Correlation VAE (NED-TCAE) in Table 1. In some application cases where only the group-wise disentanglement is needed, and the variable-wise disentanglement in \(z_e, z_f\) and \(z_g\) is not required. This kind of disentanglement can be referred to as a “Half Correlation Disentanglement” of the graphs, where the penalty for Term \(\mathcal{C}\) is ignored, leading to another variant model NED-HCVAE, as defined in Table 1.

When calculating Term \(\mathcal{A}\), we utilize the Naïve Monte Carlo approximation based on a mini-batch of samples to underestimate \(q_\phi(Z), q_\phi(z_e), q_\phi(z_f),\) and \(q_\phi(z_g),\) as described in work proposed by Chen et al. [6].

4.2.3 Generalization of variable-wise disentanglement \(\mathcal{C}\). To further enforce the variable-wise disentanglement, we generalize Term \(\mathcal{C}\) by decomposing it to obtain the “Variable Total Correlation” (VTC) terms to largely enforce the variable-wise disentanglement in \(z_e, z_f\) and \(z_g\) respectively. The following shows the decomposition of \(D_{KL}(q_\phi(z_f)\|p(z_f))\) in Term \(\mathcal{C}\) as an example:

\[
-D_{KL}(q_\phi(z_f)\|p(z_f))
\]

\[
= -E_{q_\phi(z_f)}[\log \frac{q_\phi(z_f)}{p(z_f)} + \log \frac{\prod_d q_\phi(z^{(d)}_f)}{\prod_d p(z^{(d)}_f)} + \log \frac{p(z_f)}{p(z_f)}]
\]

\[
= -E_{q_\phi(z_f)}[\log \frac{q_\phi(z_f)}{p(z_f)} - \log \frac{p(z_f)}{p(z_f)} - \sum_d D_{KL}(q_\phi(z^{(d)}_f)\|p(z^{(d)}_f))]
\]

Thus, enforcing them can help ensure the mutual information between each types of latent representations and observed graphs.

4.2.4 Generalization of conditional distribution Term \(\mathcal{B}\). In some cases, we are really only concerned with node attributes or edge attributes, so we need only control either the nodes or edges when generating the graph. Thus, to learn the types of factors involved, we can anchor a single group of latent variable (e.g., \(z_e\)), to yield higher mutual information with the observation graphs \(G\).

First, if we decompose Term \(\mathcal{B}\) in Eq. 7, we have:

\[
\begin{align*}
& -\log \frac{q_\phi(z_e)q_\phi(z_f)|E} {q_\phi(z_e)q_\phi(z_f)q_\phi(z_g)} \\
& = \log \frac{q_\phi(z_e)} {q_\phi(z_e)|E} + \log \frac{q_\phi(z_f)q_\phi(z_g)} {q_\phi(z_f)|E} + \log \frac{q_\phi(z_g)} {q_\phi(z_g)|E}
\end{align*}
\]

Since each of the above three terms actually represents mutual information between observations and latent representations, because

\[
-\log \frac{q_\phi(z_e)} {q_\phi(z_e)|E} = -\log \frac{q_\phi(z_e)q_\phi(z_f)q_\phi(z_g)} {q_\phi(z_e)q_\phi(z_f)q_\phi(z_g)|E} = \log \frac{q_\phi(z_e)q_\phi(z_f)q_\phi(z_g)|E} {q_\phi(z_e)q_\phi(z_f)q_\phi(z_g)} = I(z_e, E)
\]

Thus, enforcing them can help ensure mutual information between each type of latent representations and observed graphs. The extensive model that enforces either of the three terms is named as NED-AnchorVAE in Table 1.

4.2.5 Relation to existing models. Next, we demonstrate that the existing disentanglement methods, where only the disentanglement representation learning of nodes attributes is considered, are actually special cases of our proposed new frameworks.

First, as a special case of attributed graph, image only involves node attributes and node-related factors matters. Hence in this special case, the NED-VAE objective can be rewritten by ignoring \(z_c\) and \(z_g\) as:

\[
\mathcal{L}(\theta, \phi, G, Z, \beta) = D_{KL}(q_\phi(Z|F)|p(z_f)) + \beta D_{KL}(q_\phi(z_f|F)|p(z_f)),
\]

which is the same objective as that defined in \(\beta = VA\) [22] for the image domain. In the same way, we can easily demonstrated that DIP-VA [27] is a special case of the the proposed NED-IPVAE-I, obtained by enforcing the inferred priors disentanglement, and InforVAE is a special case of the proposed NED-IPVAE-II.
In addition, the proposed NED-TCVAE is a more general form that includes the objective of two existing methods FactorVAE [23] and β-TCVAE [6] which share the same objectives. For example, when the weight β of Term A is 0, and there is no z_e and z_g, there is no need to enforce the group-wise disentanglement among the edge-latent z_e, node-related latent z_f, and node-edge joint latent z_g. Only the variable-wise disentanglement C_f is used.

4.3 Complexity Analysis
The proposed NED-VAE requires O(N^2) operations in time complexity and O(N^2) computation complexity in terms of number of nodes in the graph, which pave the way toward modest scale graphs with hundreds or thousands of nodes, compared to most of the existing graph generation methods, which often have O(N^3) or even O(N^4) computational costs. For example, GraphVAE [43] requires O(N^4) operations in the worst case and Li et al. [32] uses graph neural networks to perform a form of message passing with O(MN^2) operations to generate a graph.

5 EXPERIMENT
This section reports the results of both qualitative and quantitative experiments that are carried out to test the performance of NED-VAE and its extensions on two synthetic and one real-world datasets. All experiments are conducted on a 64-bit machine with an NVIDIA GPU (GTX 1070, 1683 MHz, 16 GB GDDR5).

5.1 Dataset
5.1.1 Erdos-Renyi Graphs. Erdos-Renyi (ER) graphs are generated based on three types of factor. One is an edge-related factor a that refers to the probability of edge creation in a graph following the rule specified in [13]: the second is a node-related factor b which is the mean of a Gaussian random distribution (the standard is set to 0.1), based on which node attribute F_{i,1} is generated; and the third is a node-edge-joint related factor c defining the function: F_{i,2} = degree(i) + 10 \ast c (where c is a positive integer chosen from 1 to 10), based on which the second node attribute F_{i,2} is generated. Here, degree(i) refers to the degree of Node i. The dimension of the node attribute and edge attribute is 2 and 1 respectively. A total of 25,000 ER graphs are used for training and 12,500 for testing.

5.1.2 Watts Strogatz Graphs. Watts Strogatz (WR) graphs are also generated based on three types of factor. One is an edge-related factor a that indicates the number of nearest neighbours that each node is joined to in a ring topology [47]: the second is a node-related factor b that refers to the mean of a Gaussian distribution (the standard is set as 0.01) based on which node attribute is generated; and the third factor is a node-edge-joint related factor c that not only defines the probability of rewiring each edge for graph topology but also defines the second node attribute as F_{i,2} = degree(i) + 10 \ast c. The dimension of the node attribute and edge attribute is 2 and 1 respectively. A total 25,000 WR graphs used for training and 12,500 for testing.

5.1.3 Protein Structure Dataset. Protein structures can be formulated as graph structured data where each amino acid is a node and the geo-spatial distances between them are edges. To generate the dataset, we simulate the dynamic folding process of a protein peptide with a sequence AGAAAAGA, which for our purposes can be considered as a graph of 8 nodes with node attributes (x, y, z) corresponding to 3D coordination of the C_a atom of each amino acid. The protein contact map (graph topology) is generated based on fully atomistic molecular dynamics simulations. There are two factors involved in generating the contact maps and nodes attributes: simulation time (T) and ionic concentration (C), both of which are edge-related factors. Here, 38 values are used for the ionic concentration (C) and 2,000 values are used for the simulation time (T) to generate the dataset, producing 38,000 samples for training and 38,000 samples for testing.

5.2 Comparison Methods
Since graphVAE [43] is the only existing method that fits the requirement of graph disentanglement (i.e., not only learning the representations of graphs but also generate both edge and node attributes), it is utilized as one comparison method. In addition, to validate the necessities of inferring three types of representations separately, a baseline model called GDVAE is used, which has only one graph encoder for inferring an overall graph representation vector. The proposed model NED-VAE as well as the extensions (except NED-AnchorVAE) in Table 1 are all tested and compared.

5.3 Evaluation Metrics
5.3.1 Qualitative Metrics. As it is important to be able to measure the level of disentanglement achieved by different models, we search to qualitatively demonstrate that our proposed NED-VAE model and its extensions consistently discover more latent factors and disentangles them in a cleaner fashion than the previous models. By learning a latent code representation of a graph, we assume that each variable in the latent code corresponds to a certain factor or property that is used to generate the graphs’ edge and node attributes. Thus, by changing the value of one variable continuously and fixing the remaining variables, we can visualize the corresponding change in the generated graphs.

5.3.2 Quantitative Metrics. We used four quantitative metrics to evaluate the disentanglement of the proposed models. β – M [22] measures disentanglement by examining the accuracy of a linear classifier that predicts the index of a fixed factor of variation; while F – M [23] addresses several issues by using a majority vote classifier on a different feature vector that represents a corner case in the β – M; and the modularity score (mod) [39] measures whether each dimension of z depends on at most a factor describing the maximum variation using their mutual information. Finally, disentanglement metric, DCI metric [12] computes the entropy of the distribution obtained by normalizing the importance of each dimension of the learned representation for predicting the value of a factor of variation. All the implementation details are the same as those in the work proposed by Locatello et al. [34].

5.4 Results for ER dataset
5.4.1 Qualitative Evaluation. For ER graphs visualization, the color of nodes is used to represent the value of the node-related factor b, and graph topology is used to represent the value of the edge-related
factor $a$, and the size of the node is used to represent the value of the edge-node-combined factor $c$. The values of the latent variables range in $[0, 10]$ and some segments of the generated graphs is shown in Fig. 3. All of the proposed node-edge disentanglement models (NED-) shows the best capabilities in discovering and disentangling all the three types of factors than the graphVAE and the baseline GVAE. For example, the node-related factor travels well with the obvious color ranging, while the discovered node-related factor by graphVAE is not disentangled well because it has some influence on the edges. This is highly due to the powerful co-decoder in the generation of both nodes.

Table 2: Comparison of disentanglement scores of the proposed NED-VAE and its extensions for three datasets.

| Dataset | method       | $β$-M(%) | $F$-M(%) | DCI | Mod |
|---------|--------------|----------|----------|-----|-----|
| ER      | GraphVAE     | 79.20    | 33.30    | 0.33 | 0.75 |
|         | GDVAE        | 79.20    | 33.34    | 0.33 | 0.74 |
|         | NED-VAE      | 97.20    | 86.70    | 0.62 | 0.95 |
|         | NED-IPVAE-I  | 99.71    | 98.84    | 0.73 | 0.92 |
|         | NED-IPVAE-II | 99.90    | 98.70    | 0.71 | 0.93 |
|         | NED-TCVAE    | 96.70    | 88.00    | 0.64 | 0.92 |
|         | NED-VTCVAE   | 94.00    | 59.10    | 0.63 | 0.97 |
| WS      | GraphVAE     | 73.10    | 37.87    | 0.13 | 0.49 |
|         | GDVAE        | 73.06    | 37.86    | 0.13 | 0.62 |
|         | NED-VAE      | 100.00   | 64.96    | 0.16 | 0.52 |
|         | NED-IPVAE-I  | 99.30    | 91.23    | 0.16 | 0.50 |
|         | NED-IPVAE-II | 100.00   | 97.82    | 0.16 | 0.50 |
|         | NED-TCVAE    | 94.91    | 64.70    | 0.16 | 0.50 |
|         | NED-VTCVAE   | 94.50    | 49.33    | 0.17 | 0.51 |
| Protein | GraphVAE     | 63.42    | 61.67    | 0.31 | 0.69 |
|         | GDVAE        | 60.46    | 55.20    | 0.31 | 0.67 |
|         | NED-VAE      | 60.00    | 64.00    | 0.28 | 0.67 |
|         | NED-IPVAE-I  | 57.63    | 50.25    | 0.25 | 0.68 |
|         | NED-VTCVAE   | 58.40    | 50.00    | 0.24 | 0.67 |

5.4.2 Quantitative Evaluation. Four quantitative evaluation metrics are tested on WR dataset on different models and compared in Table 2. The proposed node-edge disentanglement models all shows greater superiority than graphVAE and baseline GVAE. Specifically, NED-VAE and NED-IPVAE-I both achieve 100% in $β − M$, outperforming comparison methods by 26.9% and other proposed extensions by 3.9%. NED-IPVAE-II achieves 97.8% score in $F − M$, outperforming comparison methods by 60.3% and other proposed extensions by 30.8%.

5.6 Results for Protein Structure dataset

5.6.1 Qualitative Evaluation. We evaluate the control of the factor of simulation time $(T)$ to the generation of edges by visualizing the contact map of the proteins. The value of the relevant latent variables ranges in $[0, 10]$ and some segment of the generated contact maps are shown in Fig. 5. All of the proposed models are capable of finding $T$ factor, while graphVAE shows bad performance in a very slight variation of structure. In addition, qualitative evaluation on protein dataset is also meaningful in analyzing how the proteins folds (reflected in contact maps) as the time flies.

5.6.2 Quantitative Evaluation. Four quantitative evaluation metrics are also tested on protein dataset on different models and compared in Table 2. The proposed node-edge disentanglement models, especiallly NED-VAE all shows greater superiority than graphVAE and baseline GVAE. Specifically, NED-VAE outperforms the comparison methods by 14.9%, 32.2%, and 13.1% on metrics of $β − M$, DCI and Modularity respectively; and outperforms other proposed extensions by 6.8%, 17.2%, and 2.8% on metrics of $β − M$, DCI and Modularity respectively. This proves that the proposed NED-VAE still have superiority even when there is only edge-related factor.

6 CONCLUSION

We have introduced NED-VAE, a novel and the first method for disentangling on attributed graphs as far as we know. Moreover, we propose a generic framework of objectives including various derived disentanglement penalties to solve different issues in dealing with graph structured data, such as group-wise and variable-wise disentanglement; multiple trade-off issues between reconstructed edges and nodes, and edge-related, node-related, and node-edge-related latent. Finally, we have performed an experimental evaluation of disentangling qualitatively and quantitatively for the proposed NED-VAE and its extensions. The comparison with graphVAE and a baseline model validates the effectiveness of the graph disentanglement architecture and the necessities of separately learning three types of latent representations.
ACKNOWLEDGMENTS

This work was supported by the National Science Foundation (NSF) Grant No. 1755850, No. 1841520, No. 1907805, No. 1763233, a Jeffress Memorial Trust Award, NVIDIA GPU Grant, and Design Knowledge Company (subcontract number: 10827.002.120.04). Y. Ye’s work was partially supported by the NSF Grants IIS-2027127, IIS-1951504, CNS-1940855, CNS-1946327, CNS-1814825, OAC-1940855, and the NIJ 2018-75-CX-0032. This material is additionally based upon work supported by (while serving at) the NSF. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.
Figure 5: Generated contact maps from different models when one edge-related latent variable ranges from 0 to 10 in protein dataset: more blank spaces indicates higher degree of protein folding.

REFERENCES

[1] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. ICLR (2017).

[2] Yunsheng Bai, Hao Ding, Yang Qiao, Agustin Marinovic, Ken Gu, Ting Chen, Yizhou Sun, and Wei Wang. 2019. Unsupervised inductive graph-level representation learning via graph-graph proximity. In Proceedings of the 28th International Joint Conference on Artificial Intelligence. AAAI Press, 1988–1994.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence 35, 8 (2013), 1798–1828.

[4] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zügner, and Stephan Günnemann. 2018. NetGAN: Generating Graphs via Random Walks. In ICML. 699–710.

[5] Ioan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. ICLR (2013).

[6] Tian Q Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. 2018. Isolating diffusion mechanisms with the f-statistic loss. In ICLR. 6114–6125.

[7] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. ICLR 2, 5 (2017), 6.

[8] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of intertwinement learning via graph-graph proximity. In Proceedings of the 28th International Joint Conference on Artificial Intelligence. AAAI Press, 1988–1994.

[9] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2005. The graph Buddy model for generating realistic graphs. In ICALP. 729–734.

[10] Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter Battaglia. 2018. Learning deep generative models of graphs. (2018).

[11] Yanbei Liu, Xiao Wang, Shu Wu, and Zhitao Xiao. 2019. Independence Promoted Graph Disentangled Networks. (2019).

[12] Xiaojie Guo, Liang Zhao, and et al. 2019. Deep Multi-attributed Graph Translation. In ICDM. Vol. 3. IEEE, 2434–2444.

[13] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005. Vol. 2. IEEE, 729–734.

[14] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. (2017).

[15] Alexander LeClair, Sakib Hasque, Lingfei Wu, and Collin McMillan. 2020. Improved disentangled representations for recommendation. In NeurIPS. 5712–5723.

[16] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. (2017).

[17] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. The graph Buddy model for generating realistic graphs. In ICALP. 729–734.

[18] Xiaojie Guo, Lingfei Wu, and Liang Zhao. 2018. Deep graph translation. arXiv preprint arXiv:1805.09980 (2018).

[19] Xiaojie Guo, Liang Zhao, and et al. 2019. Deep Multi-attributed Graph Translation with Node-Edge Co-evolution. In ICDM.

[20] Yu Chen, Lingfei Wu, and Mohammed J Zaki. 2020. Graphflow: Exploiting conversation flow with graph neural networks for conversational machine comprehension. IJCAI (2020).

[21] Tian Qi Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. 2018. Isolating diffusion mechanisms with the f-statistic loss. In ICLR. 6114–6125.

[22] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. ICLR 2, 5 (2017), 6.

[23] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by factorising. In NeurIPS. 5712–5723.

[24] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. (2013).

[25] Diederik P Kingma and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. ICLR (2016).

[26] Babak Esmaeili, Hao Wu, Sarthak Jain, and et al. 2019. Structured Disentangled Generative Parameters of Random and Real-World Graphs. In NeurIPS. 729–734.

[27] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. In NeurIPS. 5712–5723.

[28] Thomas N Kipf and Max Welling. 2016. Variational auto-encoders. (2016).

[29] Alexander LeClair, Sakib Hasque, Lingfei Wu, and Collin McMillan. 2020. Improved disentangled representations for recommendation. In NeurIPS. 5712–5723.

[30] Sachin Agarwal, Yujia Li, Oriol Vinyals, and Christopher B. Bell. 2018. Hierarchical Attention Based Spatial-Temporal Graph-to-Sequence Learning for Graphs. In IJCAI. 2018.

[31] Xiaoji Guo, Jiaxin Sun, Huaxin Shao, Chaochao Ai, and Shangxian Zeng. 2019. Large-scale Cost-aware Classification Using Feature Disentangled Networks. In ICDM. 1975–1985.

[32] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic variational inference. In AISTATS. 2434–2444.

[33] Francesco Locatello, Stefan Bauer, Mohammad Jahani, Maja Pantic, and Bernhard Schölkopf. 2019. Learning deep generative models of graphs. In ICDM. Volume 32–II–1278.

[34] Karl Riveray and Michael C Mozer. 2018. Learning deep disentangled embeddings with the f-statistic loss. In NeurIPS. 185–194.

[35] Romain Lopez, Jeffrey Regier, Michael I Jordan, and Nir Yosef. 2018. Information constraints on auto-encoding variational bayes. In NeurIPS. 6114–6125.

[36] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic variational inference. In AISTATS. 2434–2444.

[37] Babak Esmaeili, Hao Wu, Sarthak Jain, and et al. 2019. Structured Disentangled Generative Parameters of Random and Real-World Graphs. In NeurIPS. 729–734.

[38] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. The graph Buddy model for generating realistic graphs. In ICALP. 729–734.

[39] Martin Simonovsky and Nikos Komodakis. 2018. Graphvae: Towards generation of small graphs using variational autoencoders. In ICML. 6114–6125.

[40] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. ICLR 2, 5 (2017), 6.

[41] Tian Q Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. 2018. Isolating sources of disentanglement in variational autoencoders. In NeurIPS. 2610–2620.

[42] Yu Chen, Lingfei Wu, and Mohammad J Zaki. 2020. GraphFlow: Exploiting conversation flow with graph neural networks for conversational machine comprehension. IJCAI (2020).

[43] Yu Chen, Lingfei Wu, and Mohammad J Zaki. 2020. Reinforcement learning based graph-to-sequence model for natural question generation. ICLR (2020).

[44] Song Han, Li-Jia Li, Kang He, personalised neural networks. In IJCAI. 2017.

[45] Kai Shen, Lingfei Wu, Fangli Xu, Siliang Tang, Jun Xiao, and Yueting Zhuang. 2020. Deep Multi-attributed Graph Translation. In ICDM. Vol. 2. IEEE, 729–734.

[46] Xiaojie Guo, Lingfei Wu, and Liang Zhao. 2018. Deep graph translation. arXiv preprint arXiv:1805.09980 (2018).

[47] Xiaojie Guo, Liang Zhao, and et al. 2019. Deep Multi-attributed Graph Translation with Node-Edge Co-evolution. In ICDM.

[48] Michael Defferrard, Xavier Bresson, and et al. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In NeurIPS. 3844–3852.

[49] Ian Goodfellow, Jean Pouget-Abadie, and et al. 2014. Generative adversarial nets. In NeurIPS. 2672–2680.

[50] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, Vol. 1. IEEE, 5712–5723.

[51] Xiaojie Guo, Lingfei Wu, and Liang Zhao. 2018. Deep graph translation. arXiv preprint arXiv:1805.09980 (2018).