GraphEBM: Molecular Graph Generation with Energy-Based Models

Meng Liu 1  Keqiang Yan 1  Bora Oztekin 1  Shuiwang Ji 1

Abstract
Molecular graph generation is an emerging area of research with numerous applications. This problem remains challenging as molecular graphs are discrete, irregular, and permutation invariant to node order. Notably, most existing approaches fail to guarantee the intrinsic property of permutation invariance, resulting in unexpected bias in generative models. In this work, we propose GraphEBM to generate molecular graphs using energy-based models. In particular, we parameterize the energy function in a permutation invariant manner, thus making GraphEBM permutation invariant. We apply Langevin dynamics to train the energy function by approximately maximizing likelihood and generate samples with low energies. Furthermore, to generate molecules with a specific desirable property, we propose a simple yet effective strategy, which pushes down energies with flexible degrees according to the properties of corresponding molecules. Finally, we explore the use of GraphEBM for generating molecules with multiple objectives in a compositional manner. Comprehensive experimental results on random, goal-directed, and compositional generation tasks demonstrate the effectiveness of our proposed method.

1. Introduction
A fundamental problem in drug discovery and material science is to find novel molecules with desirable properties. One popular method is to search in the chemical space based on molecular property prediction (Gilmer et al., 2017; Wu et al., 2018; Yang et al., 2019; Stokes et al., 2020; Wang et al., 2020). Recently, molecular graph generation has provided an alternative and promising way for this problem by directly generating desirable molecules, thus circumventing the expensive search of the chemical space. Despite intensive efforts recently, molecular graph generation remains challenging since the underlying graphs are discrete, irregular, and permutation invariant to node order.

As noted in Section 2.1, existing approaches have achieved promising success by generating molecular graphs based on various generative methods, including variational autoencoders (VAEs) (Kingma & Welling, 2013), generative adversarial networks (GANs) (Goodfellow et al., 2014), flow models (Dinh et al., 2014; Rezende & Mohamed, 2015) and recurrent neural networks (RNNs). However, most of them fail to preserve the intrinsic property of permutation invariance, which might yield different likelihoods for different permutations of the same graph.

In this work, we propose GraphEBM to generate molecular graphs with energy-based models (EBMs) (LeCun et al., 2006). Since our parameterized energy function is permutation invariant, we can show that our GraphEBM preserves the permutation invariance property. We use Langevin dynamics (Welling & Teh, 2011) to train the energy function by maximizing likelihood approximately and generate samples from the trained energy function. To our knowledge, our GraphEBM is the first energy-based model that can generate attributed molecular graphs. Furthermore, in order to generate molecules with a specific desirable property, we propose a novel, simple, and effective strategy to train our GraphEBM for goal-directed generation by pushing down energies with flexible degrees according to the property values of corresponding molecules. Significantly, we show that GraphEBM can generate molecules with multiple objectives in a compositional manner, which cannot be achieved by any existing methods. This provides a new and promising way for multi-objective molecule generation.

We conduct experiments on a variety of common benchmarks for molecule generation. Results show that GraphEBM can generate realistic molecules, which is significant given that EBMs have rarely been explored in the graph domain. We also conduct experiments to demonstrate the effectiveness of our proposed method for goal-directed generation. Finally, experimental results on compositional generation show that our GraphEBM has the potential to generate molecules with multiple objectives in a compositional manner.
Table 1. Summary and comparison of existing molecular graph generation methods.

| Method             | Generative method | Generation process | Permutation invariance | Compositional generation |
|--------------------|-------------------|--------------------|------------------------|--------------------------|
|                     | VAE   | GAN   | Flow | RNN | EBM | One-shot | Sequential |                |                          |
| GraphVAE (Simonovsky & Komodakis, 2018) | ✓     | -     | -    | ✓   | ✓   | ✓        | ✓          |                | ✓                        |
| DeepGMG (Li et al., 2018) | -     | -     | ✓    | -   | ✓   | -        | -          |                | -                        |
| CGVAE (Liu et al., 2018) | ✓     | -     | -    | ✓   | ✓   | -        | -          |                | -                        |
| MoVAE (De Cao & Kipf, 2018) | ✓     | ✓     | -    | -   | ✓   | ✓        | -          |                | -                        |
| RVE (Ma et al., 2018) | ✓     | ✓     | -    | -   | -   | -        | -          |                | -                        |
| GCNP (You et al., 2018) | -     | ✓     | -    | -   | ✓   | ✓        | -          |                | -                        |
| JT-VAE (Jin et al., 2018) | ✓     | ✓     | -    | -   | ✓   | ✓        | -          |                | -                        |
| MolecularRNN (Popova et al., 2019) | -     | -     | ✓    | -   | -   | -        | -          |                | -                        |
| GraphNVP (Madhawa et al., 2019) | -     | -     | ✓    | -   | ✓   | ✓        | -          |                | -                        |
| Bresson & Laurent (2019) | ✓     | -     | -    | ✓   | ✓   | ✓        | -          |                | -                        |
| GRF (Honda et al., 2019) | -     | ✓     | -    | -   | ✓   | ✓        | -          |                | -                        |
| GraphAF (Shi et al., 2019) | -     | ✓     | -    | -   | ✓   | ✓        | -          |                | -                        |
| HierVAE (Jin et al., 2020a) | ✓     | -     | -    | -   | ✓   | ✓        | -          |                | -                        |
| MoFlow (Zang & Wang, 2020) | -     | ✓     | -    | -   | ✓   | ✓        | -          |                | -                        |
| GraphCNF (Lippe & Gavves, 2020) | -     | ✓     | -    | -   | ✓   | ✓        | -          |                | -                        |

2. Preliminary and Related Work

2.1. Molecular Graph Generation

Since molecules can be represented as SMILES strings (Weininger, 1987), early studies generate molecules based on SMILES strings, such as CVAE (Gómez-Bombarelli et al., 2018), GVAE (Kusner et al., 2017), and SD-VAE (Dai et al., 2018). Recent studies mostly represent and generate molecules as graphs (Simonovsky & Komodakis, 2018; De Cao & Kipf, 2018; Madhawa et al., 2019). We can categorize existing molecular graph generation methods based on the underlying generative methods or the generation processes. Current molecular graph generation approaches can be grouped into four categories according to their underlying generative models, i.e., VAEs, GANs, flow models, and RNNs. They can also be classified into two primary types based on their generation processes; those are, sequential generation and one-shot generation. The sequential process generates nodes and edges in a sequential order by adding nodes and edges one by one. The one-shot process generates all nodes and edges at one time.

To facilitate comparison, we summarize existing methods in Table 1. We can observe that most of them fail to satisfy an intrinsic property of graphs; that is, permutation invariance. Specifically, a generative model should yield the same likelihood for different permutations of the same graph. Currently, permutation invariance remains to be a challenging goal to achieve. The sequential generation approaches have to choose a specific order of nodes, thus failing to preserve permutation invariance. Among the one-shot methods, Bresson & Laurent (2019) also use the specific node order given by the SMILES representation. GraphVAE and RVAE perform an approximate and expensive graph matching to train the VAE model, and they cannot achieve exact permutation invariance. MolGAN circumvents this issue by using a likelihood-free method. The recent one-shot flow methods have the potential to satisfy this property. However, GraphNVP, GRF, and MoFlow cannot preserve this property since the masking strategies in the coupling layers are sensitive to node order. An exception is GraphCNF, which achieves permutation invariance by assigning likelihood independent of node ordering via categorical normalizing flows.

In this work, we propose to develop energy-based models (EBMs) (LeCun et al., 2006) for molecular graph generation. EBMs are a class of powerful methods for modeling richly structured data, but their use for graph generation has been under-explored. We show that our method can achieve the desirable property of permutation invariance. Additionally, our method has the potential to generate molecules in a compositional manner towards multiple objectives, which cannot be achieved by any existing methods.

2.2. Energy-Based Models

Modeling variables by defining an unnormalized probability density has been explored for decades (Hopfield, 1982; Ackley et al., 1985; Cipra, 1987; Dayan et al., 1995; Cipra, 1987; Zhu et al., 1998; Hinton, 2012). Such methods are known and unified as energy-based models (EBMs) (LeCun et al., 2006) in machine learning. EBMs capture the dependencies of variables by assigning a scalar energy to each configuration of the variables with a learnable energy function. Given a trained EBM, inference is to find the configurations that yield low energies. Training an EBM aims at obtaining an energy function where observed configurations are associated with lower energies than unobserved ones.

EBMs can be naturally used as generative models since data points near the underlying data manifold are assigned lower energies than other data points, which defines an unnormal-
We formally represent a molecular graph as $G$, a method that is similar to EBMs. How-ever, these two methods can only generate graph structures and it is not straightforward to use them on attributed graphs. In this work, we propose GraphEBM to generate attributed molecular graphs using EBMs. Hence, we consider GraphEBM to be the first energy-based model capable of generating attributed graphs.

3. The Proposed GraphEBM

In this section, we present GraphEBM by describing the internal structure of the parameterized energy function (Section 3.2), showing that GraphEBM satisfies the desirable property of permutation invariance (Section 3.3), and describing the training (Section 3.4) and generation (Section 3.5) process of GraphEBM. Then, we introduce our proposed strategy for goal-directed generation based on GraphEBM (Section 3.6). Finally, we explore the potential of compositional generation using GraphEBM (Section 3.7).

3.1. Problem Formulation

Molecules can be naturally represented as graphs by considering atoms and bonds as nodes and edges, respectively. We formally represent a molecular graph as $G = (X, A)$, where $X$ is the node feature matrix and $A$ is the adjacency tensor. Let $k$ be the number of nodes in the graph, $b$ and $c$ denote the number of possible types of nodes and edges, respectively. Then we have $X \in \{0,1\}^{k \times b}$ and $X_{i,p} = 1$ if node $i$ belongs to type $p$, $A \in \{0,1\}^{k \times k \times c}$ and $A_{(i,j,q)} = 1$ denotes that an edge with type $q$ exists between node $i$ and node $j$. Following Madhawa et al. (2019); Zang & Wang (2020), we let $n$ denote the maximum number of atoms that a molecule has in a given dataset. We insert virtual nodes into molecular graphs that have less than $n$ nodes such that the dimensions of $X$ and $A$ keep the same for all molecules. Also, for any two nodes that are not connected in the molecule, we add a virtual edge between them. We can consider the virtual node and the virtual edge as an additional node type and edge type, respectively. Hence, for all molecules in a certain dataset, $X \in \{0,1\}^{n \times (b+1)}$ and $A \in \{0,1\}^{n \times n \times (c+1)}$.

Naturally, a distribution of molecular graphs can be denoted as $p(X, A)$. Since permutation invariance is a fundamental property of graphs, an ideal generative model should associate the same probability to various permutations of the same molecular graph. Formally, an ideal generative model is expected to satisfy

$$p(X, A) = p(X^\pi, A^\pi),$$

where $\pi$ denotes any permutation of node order. For simplicity, we use the superscript $\pi$ to denote that the corresponding matrix or tensor is arranged according to the node order given by $\pi$.

3.2. Parameterized Energy Function

Following the above notations, the energy function for molecular graphs can be denoted as $E_\theta(X, A)$. Specifically, we model $E_\theta(X, A)$ by a graph neural network, where $\theta$ denotes parameters in the network. Many deep learning methods on graphs have been proposed and have achieved great success in many tasks, such as node classification (Kipf & Welling, 2017; Hamilton et al., 2017; Monti et al., 2017; Veličković et al., 2018; Gao et al., 2018; Xu et al., 2018; Klicpera et al., 2019; Wu et al., 2019; Liu et al., 2020b; Jin et al., 2020c; Chen et al., 2020; Liu et al., 2020c; Hu et al., 2020), graph classification (Zhang et al., 2018; Ying et al., 2019; Liu et al., 2020b; You et al., 2020; Gao et al., 2020), and link prediction (Zhang & Chen, 2018; Cai et al., 2020). In this work, we use a variant of relational graph convolutional networks (R-GCN) (Schlichtkrull et al., 2018) to learn the node representations since molecular graphs have categorical edge types. Formally, the layer-wise forward-propagation is defined as

$$H^{\ell+1} = \sigma \left( \sum_{k=1}^{c+1} (A_{(i,:k)} H^{\ell} W^{\ell}_k) \right).$$

$A_{(i,:k)}$ is the $k$-th channel of the adjacency tensor. $H^{\ell}$ is the node representation matrix at layer $\ell$. $W^{\ell}_k$ represents the trainable weight matrix for edge type $k$ at layer $\ell$. $\sigma(\cdot)$ denotes a non-linear activation function. The initial node representation matrix $H^0 = X$. In each layer, message passing is conducted among the nodes independently for each type of edge. Then, the information is integrated together by a sum operator. We stack $L$ such layers. Hence, the final node representation matrix is $H^L \in \mathbb{R}^{n \times d}$, where
We have the parameterized energy function $W$ defined as

$$W = \sum_{i=1}^{n} H_{i;i}^L \in \mathbb{R}^d.$$  (4)

Finally, the scalar energy associated with the molecular graph can be obtained by applying a transformation as

$$E = h_G^T W \in \mathbb{R},$$  (5)

where $W \in \mathbb{R}^d$ is the trainable parameters.

### 3.3. Permutation Invariance

Permutation invariance is an intrinsic and desirable inductive bias for graph modeling. We note that our proposed GraphEBM satisfies this fundamental property due to our permutation invariant energy function. Specifically, each layer of our graph neural network in Eq. (3) is permutation equivariant. In addition, the readout operation in Eq. (4) is permutation invariant. Therefore, our parameterized energy function is permutation invariant thus satisfying

$$E_\theta(X, A) = E_\theta(X^\pi, A^\pi).$$  (6)

According to Eq (1), the energy function defines a distribution over data. Specifically, the likelihood is proportional to the negative exponential of the corresponding energy. Hence, based on Eqs (6) and (1), we can obtain

$$p_\theta(X) = p_\theta(X^\pi).$$  (7)

Thus, our GraphEBM can preserve permutation invariance by modeling graphs in a permutation invariant manner.

### 3.4. Training

We have the parameterized energy function $E_\theta(X, A)$ defined in Section 3.2. The natural follow-up question is how to effectively train the parameterized energy function.

Intuitively, a good energy function should assign lower energies to data points that correspond to real molecular graphs and higher energies to other data points. Hence, a straightforward idea is to train the parameterized energy function by maximizing the likelihood of real data defined in Eq. (1). Let $p_D$ be the distribution of the real data. We can achieve maximum likelihood by minimizing the negative log likelihood of real data. Formally,

$$\mathcal{L}_{ML} = E_{(X,A) \sim p_D} [-\log p_\theta(X, A)],$$  (8)

where $-\log p_\theta(X, A) = E_\theta(X, A) + \log Z(\theta)$ according to Eq. (1). It has been shown (Hinton, 2002; Turner, 2005) that the objective in Eq. (8) has the below gradient:

$$\nabla_\theta \mathcal{L}_{ML} = E_{(X^\oplus, A^\oplus) \sim p_D} \left[ \nabla_\theta E_\theta(X^\oplus, A^\oplus) \right] - E_{(X^\ominus, A^\ominus) \sim p_\theta} \left[ \nabla_\theta E_\theta(X^\ominus, A^\ominus) \right].$$  (9)

Intuitively, this gradient pushes down the energies of positive samples $(X^\oplus, A^\oplus)$ and pushes up the energies of samples $(X^\ominus, A^\ominus)$ that are sampled from $p_\theta$. As defined in Eq. (1), $p_\theta$ is the distribution given by the energy function. Following Du et al. (2020b), we refer to $(X^\ominus, A^\ominus)$ as hallucinated samples. In practice, however, sampling $(X^\ominus, A^\ominus)$ from $p_\theta$ is challenging, since $Z(\theta)$ in Eq. (1) is intractable.

To overcome this issue, we follow Du & Mordatch (2019) to sample $(X^\ominus, A^\ominus)$ from an approximated $p_\theta$ using Langevin dynamics (Welling & Teh, 2011). Particularly, a sample $(X^\ominus, A^\ominus)$ is initialized randomly and refined iteratively by

$$X^k = X^{k-1} - \frac{\lambda}{2} \nabla_X E_\theta(X^{k-1}, A^{k-1}) + w^k,$$

$$A^k = A^{k-1} - \frac{\lambda}{2} \nabla_A E_\theta(X^{k-1}, A^{k-1}) + \eta^k,$$

$$w^k, \eta^k \sim \mathcal{N}(0, \sigma^2),$$  (10)

where $k$ denotes the iteration step, $\frac{\lambda}{2}$ is the step size, and $w^k$ and $\eta^k$ are added noise sampled from a Gaussian distribution. As demonstrated by Welling & Teh (2011), the obtained samples $(X^k, A^k)$ approach samples from $p_\theta$ as $k \rightarrow \infty$ and $\frac{\lambda}{2} \rightarrow 0$. In practice, we let $K$ denote the number of iteration steps of Langevin dynamics and use the resulting sample $(X^K, A^K)$ as $(X^\ominus, A^\ominus)$ in Eq. (9).

We illustrate the training process of our GraphEBM in Figure 1. Since Langevin dynamics is for continuous data, we model the hallucinated samples by continuous format. For consistency, we can also use dequantization techniques (Dinh et al., 2016; Kingma & Dhariwal, 2018) to convert the discrete positive samples to continuous data by adding uniform noise, as shown in the left part of Figure 1. The dequantization can be formally expressed as

$$X' = X + tu; \quad u \sim [0, 1)^n \times (b+1),$$

$$A' = A + tu; \quad u \sim [0, 1)^n \times (c+1),$$  (11)

where $t \in [0, 1)$ is a scaling hyperparameter. After dequantization, we apply a normalization to the adjacency tensor, which is a common step in modern graph neural networks (Kipf & Welling, 2017). Formally,

$$A^\oplus_{(i,:;k)} = D^{-1} A'_{(i,:;k)}; \quad k = 1, \ldots, c + 1,$$  (12)

where $D$ is the diagonal degree matrix in which $D_{i,i} = \sum_j D_{i,j}$. We treat the above $A^\oplus$ and $X^\oplus = X'$ as the input for the energy function. In our case, each element of $X^\oplus$ is in $[0, 1+t]$ and each element of $A^\oplus$ is in $[0, 1]$.

Note that the above dequantization for positive samples is optional. This indicates that we can set $t = 0$ and keep...
GraphEBM: Molecular Graph Generation with Energy-Based Models

---

**Figure 1.** The training process of our GraphEBM. The left part and right part illustrate the processes of obtaining the positive sample and the hallucinated sample, respectively. The middle part shows the forward and backward propagation of the training process. In this example, $n = 9$, $k = 8$, $b = 4$, and $c = 3$. The annotations of node representation matrices denote the atom types, including carbon (C), nitrogen (N), oxygen (O), fluorine (F), and virtual atom (⋆). Note that we remove hydrogen atoms, which is a common technique in the community. The annotations of adjacency tensor indicate the bond types, including single (S), double (D), triple (T), and virtual bond (⋆). Green and purple arrows represent the forward computation of energy value for the positive sample and the hallucinated sample, respectively. The black dashed arrow denotes the gradient backpropagation.

To stabilize training, we also apply a regularization technique to the energy magnitudes. Specifically, we use the same regularization as Du & Mordatch (2019). Formally,

$$
\mathcal{L}_{\text{reg}} = E_\theta(X\oplus, A\oplus)^2 + E_\theta(X\odot, A\odot)^2.
$$

Hence, the total loss function is

$$
\mathcal{L} = \mathcal{L}_{\text{energy}} + \alpha \mathcal{L}_{\text{reg}},
$$

where $\alpha \in \mathbb{R}$ is a hyperparameter.

---

**3.5. Generation**

Let $E_\theta(X, A)$ denote the trained energy function, where $\theta^*$ represents the obtained parameters. Intuitively, if an energy function is well-shaped, the configurations with low energies should correspond to desirable molecular graphs. Hence, the generation process is to generate molecules based on the configurations $(X, A)$ that yield low energy.

An overview of the generation process is given in Figure 2. The steps are as follows. First, we initialize a data point as in Eq. (13) and then apply $K$ steps of Langevin dynamics as Eq. (10) to obtain data points that have low energy. We denote the obtained configuration as $(X^\star, A^\star)$. Second, since molecular graphs are undirected, we make the adjacency tensor to be symmetric by using $A^\star + A^\star^T$ as the new adjacency tensor. Third, we convert the continuous data to discrete ones by applying the argmax operation in the dimensions of atom types and bond types. Finally, we use validity correction introduced by Zang & Wang (2020) to refine the corresponding molecule so that the valency constraint is satisfied.
3.6. Goal-Directed Generation

We have described how our proposed GraphEBM can be used to model data distributions of molecular graphs in a permutation invariant manner and generate molecules effectively. Furthermore, for drug discovery and material design, we also need to generate molecules with desirable chemical properties. This task is termed as goal-directed generation. There are mainly three approaches in the literature for goal-directed generation. First, this task can be modeled as a conditional generation problem, where the property value can be utilized as the condition (Simonovsky & Komodakis, 2018). Second, for methods using the latent space, a predictor can be applied to learn the property value from the latent representation (Gómez-Bombarelli et al., 2018). Third, reinforcement learning can be used to optimize the properties of generated molecules (You et al., 2018). However, it is not straightforward to apply these methods to our GraphEBM for goal-directed generation since GraphEBM generates molecules implicitly using Langevin dynamics and no latent space exists. Also, using EBMs for generation that is conditional on continuous conditions is rarely studied by the community. Hence, it remains challenging to apply EBMs for goal-directed generation.

To tackle this challenge, we propose a novel, simple, and effective strategy to achieve goal-directed generation based on our GraphEBM. Our basic idea is to push down energies with flexible degrees according to the property values of corresponding molecules. If a molecule has a higher value of desirable property, we push down the corresponding energy harder. Formally, in goal-directed generation, the loss function defined in Eq. (14) becomes

\[ \mathcal{L}_{\text{energy}} = f(y)E_{\theta}(X, A^\circ) - E_{\theta}(X, A^\circ), \]

where \( y \in [0, 1] \) is the normalized property value and \( f(y) \in \mathbb{R} \) determines the degree of the push down. We use \( f(y) = 1 + e^{y} \) in this work. Thus, energies of molecules with higher property values are pushed down harder. After training, the generation process is the same as described in Section 3.5. Note that \( f(y) \) could also be a learnable function and we leave this as future work.

3.7. Compositional Generation

In addition to single property constraints, it is commonly necessary to generate molecules with multiple property constraints in drug discovery (Jin et al., 2020b). We observe that compositional generation with EBMs (Hinton, 2002), which has been shown to be effective in the image domain (Du et al., 2020a), can be naturally applied to generate molecules with multiple constraints based on our GraphEBM. Thus, we investigate compositional generation in the graph domain based on our GraphEBM.

Suppose we have two energy functions \( E_{\theta_1}(X, A) \) and \( E_{\theta_2}(X, A) \), which are trained towards two property goals respectively, as described in Section 3.6. According to Hinton (2002) and Du et al. (2020a), we can obtain a new energy function by summing the above two energy functions since the product of probabilities is equivalent to the sum of corresponding energies, according to Eq. (1). Formally,

\[ E_{\theta}(X, A) = E_{\theta_1}(X, A) + E_{\theta_2}(X, A). \]

Then we can apply the generation process described in Section 3.5 to \( E_{\theta}(X, A) \) to generate molecules towards multiple objectives in a compositional manner.

4. Experiments

4.1. Setup

Dataset. We evaluate our proposed method on two widely used molecule datasets, QM9 (Ramakrishnan et al., 2014) and ZINC250k (Irwin et al., 2012). QM9 consists of 134k organic molecules and the maximum number of atoms is 9. It contains 4 atom types and 3 bond types. ZINC250k has 250k drug-like molecules and the maximum number of atoms is 38. It includes 9 atom types and 3 edge types.

Implementation details. We kekulize molecules and remove their hydrogen atoms using RDKit (Landrum et al., 2006). In our parameterized energy function, we adopt a network of \( L = 3 \) layers with hidden dimension \( d = 64 \). We use Swish as the activation function. We set \( \alpha = 1 \) in the loss function and the standard variance \( \sigma = 0.005 \) in the gaussian noise. For training, we tune the following hyperparameters: the scale \( t \) of uniform noise \( \in [0, 1] \), the sample step \( K \) of Langevin dynamics \( \in [30, 300] \), and the step size \( \frac{\sigma^2}{2} \in [10, 50] \). All models are trained for up to 20 epochs with a learning rate of 0.0001 and a batch size of 128. It is well known that it is difficult to train EBMs. We follow the techniques adopted in Du & Mordatch (2019) to stabilize the training process. Specifically, we add spectral normalization (Miyato et al., 2018) to all layers of the network. In addition, we clip the gradient used in Langevin dynamics so that its value magnitude can be less than 0.01. GraphEBM is implemented with PyTorch (Paszke et al., 2017).
Table 2. Generation performance on ZINC250k. The results of GCPN and JT-VAE are obtained from Shi et al. (2019). The result of MoFlow is obtained by evaluating its public trained model. All other results are from their original papers.

| Method       | Validity (%) | Uniqueness (%) | Novelty (%) |
|--------------|--------------|----------------|-------------|
| GCPN         | 100.00       | 99.97          | 100.00      |
| JT-VAE       | 100.00       | 100.00         | 100.00      |
| MolecularRNN | 100.00       | 99.89          | 100.00      |
| GraphNVP     | 42.60±1.60   | 94.80±0.60     | 100.00±0.00 |
| GRF          | 73.40±0.62   | 53.7±2.13      | 100.00±0.00 |
| GraphAF      | 100.00       | 99.10          | 100.00      |
| MoFlow       | 100.00±0.00  | 99.99±0.01     | 100.00±0.00 |
| **GraphEBM** | **99.96±0.02** | **98.79±0.15** | **100.00±0.00** |

Figure 3. Visualization of molecules generated by GraphEBM.

4.2. Molecule Generation

We evaluate the ability of our proposed GraphEBM to model and generate molecules. We consider most methods reviewed in Section 2.1 as baselines. The following widely used metrics are adopted. **Validity** is the percentage of chemically valid molecules among all generated molecules. **Uniqueness** denotes the percentage of unique molecules among all valid molecules. **Novelty** corresponds to the percentage of generated valid molecules that are not present in the training set. The metrics are computed on 10,000 randomly generated molecules. Results averaged over 5 runs are reported.

The results on ZINC250k are shown in Table 2 and we include the results on QM9 in the supplement. We can observe that GraphEBM performs competitively with existing methods, which is significant considering that the study of EBMs is still in its early stage and GraphEBM is the first EBM for molecule generation. Generated samples are visualized in Figure 3, which further demonstrates that GraphEBM can generate non-trivial molecules.

To better understand the implicit generation through Langevin dynamics, we visualize this process for an example in Figure 4. We can observe that Langevin dynamics effectively refines the random initialized sample to approach a data point that corresponds to a realistic molecule.

4.3. Goal-Directed Generation

To evaluate the performance of goal-directed generation, the community usually compares the highest chemical property scores of generated molecules. We note that, due to the fact that EBMs have rarely been explored for graph generation, its performance in terms of this metric is not competitive with prior methods based on VAE, GAN, and Flow, which have been intensively studied and tuned for graph generation. We believe this could be improved with the development of this direction in the future. To empirically show the effectiveness of our goal-directed generation method proposed in Section 3.6, we train models on ZINC250k accordingly and compare the distribution of the property score between goal-directed generated molecules and random generated molecules. We consider two chemical properties, including Quantitative Estimate of Druglikeness (QED) (Bickerton et al., 2012) and penalized logP (plogp), which is the water-octanol partition coefficient penalized by the number of long cycles and synthetic accessibility.

Figure 5 (a) and (b) compare the property value distribution between goal-directed generation and random generation. It can be observed that goal-directed generation can generate more molecules with a high property value, indicating that our proposed strategy for goal-directed generation, which aims to assign lower energies to molecules with higher property values, is effective.

We further verify the effectiveness of our proposed goal-directed generation method by performing molecule optimization. That is, given a molecule, we are expected to obtain a new molecule that has a better desirable chemical property. Also, the obtained molecule should be similar to the given molecule. We adopt Tanimoto similarity of Morgan fingerprint (Rogers & Hahn, 2010) to measure the
similarity between molecules and we set the lower bound of the similarity constraint as 0.2 in this experiment. We directly use the model trained for goal-directed generation and leverage the given molecule as initialization for Langevin dynamics, instead of random initialization. Then we can obtain desirable molecules that satisfy the similarity constraint from samples given by Langevin dynamics.

Some examples on plogp optimization and QED optimization are demonstrated in Figure 5 (d) and more results are available in the supplement. This indicates that the proposed approach for goal-directed generation based on GraphEBM can perform molecule optimization effectively and the improvements on the corresponding property can be significant. More importantly and interestingly, the modifications are interpretable to some degree. Specifically, in the first example of plogp optimization shown in Figure 5 (d), our model optimizes the plogp score with a remarkable margin of 18.03 by replacing several carbon atoms with sulfur atoms, which could make the molecule more soluble in water, thus leading to a larger logP value. Additionally, plogp is highly related to the number of long cycles and synthetic accessibility. As shown in the second and third examples, our model improves the synthetic accessibility and reduces the number of long cycles by removing or breaking them. These facts indicate that our goal-directed generation method can explore the underlying chemical knowledge related to the corresponding property.

4.4. Compositional Generation

As investigated in Section 3.7, our GraphEBM has the potential to conduct compositional generation towards multiple objectives. To verify this, we combine the two energy functions obtained in goal-directed generation experiments (Section 4.3), as formulated in Eq. (18). Then we apply the generation process described in Section 3.5 to the resulting energy function to generate molecules.

The comparison of the distributions on QED and plogp between compositional generation and random generation is illustrated in Figure 5 (c). We can observe that compositional generation tends to generate more molecules with high QED and plogp scores. Additionally, the distribution of QED or plogp is similar to the corresponding distribution obtained by goal-directed generation towards a single objective (Figure 5 (a) and (b)). These facts demonstrate that our GraphEBM is able to generate molecules towards multiple objectives in a composition manner, which provides a novel and promising way for multi-objective generation.

5. Conclusion and Outlook

In this paper, we propose GraphEBM, the first energy-based model for generating molecular graphs that preserves the intrinsic property of permutation invariance. We propose to flexibly push down energies for goal-directed generation and explore to generate molecules towards multiple objectives in a compositional manner, leading to a promising method for multi-objective generation. Experimental results demonstrate that our GraphEBM can generate realistic molecules and the proposals of goal-directed generation and compositional generation are effective and promising. Since EBMs have unique advantages and have rarely been explored for graph generation, we hope our exploratory work would open the door for future research in this area.

6. Acknowledgments

We thank Yilun Du and Chengxi Zang for their helpful discussions. This work was supported in part by National Science Foundation grant DBI-1922969.
References

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

Bickerton, G. R., Paolini, G. V., Besnard, J., Muresan, S., and Hopkins, A. L. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

Bresson, X. and Laurent, T. A two-step graph convolutional decoder for molecule generation. *arXiv preprint arXiv:1906.03412*, 2019.

Cai, L., Li, J., Wang, J., and Ji, S. Line graph neural networks for link prediction. *arXiv preprint arXiv:2010.10046*, 2020.

Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, pp. 1725–1735. PMLR, 2020.

Cipra, B. A. An introduction to the ising model. *The American Mathematical Monthly*, 94(10):937–959, 1987.

Dai, H., Tian, Y., Dai, B., Skiena, S., and Song, L. Syntax-directed variational autoencoder for structured data. In *International Conference on Learning Representations*, 2018.

Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R. S. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

De Cao, N. and Kipf, T. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Du, Y. and Mordatch, I. Implicit generation and modeling with energy based models. In *Advances in Neural Information Processing Systems*, volume 32, pp. 3608–3618, 2019.

Du, Y., Li, S., and Mordatch, I. Compositional visual generation with energy based models. *Advances in Neural Information Processing Systems*, 33, 2020a.

Du, Y., Li, S., Tenenbaum, J., and Mordatch, I. Improved contrastive divergence training of energy based models. *arXiv preprint arXiv:2012.01316*, 2020b.

Gao, H. and Ji, S. Graph U-Nets. In *International Conference on Machine Learning*, pp. 2083–2092, 2019.

Gao, H., Wang, Z., and Ji, S. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1416–1424, 2018.

Gao, H., Liu, Y., and Ji, S. Topology-aware graph pooling networks. *arXiv preprint arXiv:2010.09834*, 2020.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *Proceedings of the 34th international conference on machine learning*, pp. 1263–1272, 2017.

Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pp. 1024–1034, 2017.

Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

Hinton, G. E. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pp. 599–619. Springer, 2012.

Honda, S., Akita, H., Ishiguro, K., Nakanishi, T., and Oono, K. Graph residual flow for molecular graph generation. *arXiv preprint arXiv:1909.13521*, 2019.

Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Hu, F., Zhu, Y., Wu, S., Huang, W., Wang, L., and Tan, T. Graphair: Graph representation learning with neighborhood aggregation and interaction. *Pattern Recognition*, pp. 107745, 2020.

Irwin, J. J., Sterling, T., Mysinger, M. M., Bolstad, E. S., and Coleman, R. G. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768, 2012.
GraphEBM: Molecular Graph Generation with Energy-Based Models

Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, pp. 2323–2332, 2018.

Jin, W., Barzilay, R., and Jaakkola, T. Hierarchical generation of molecular graphs using structural motifs. In *International Conference on Machine Learning*, 2020a.

Jin, W., Barzilay, R., and Jaakkola, T. Multi-objective molecule generation using interpretable substructures. In *International Conference on Machine Learning*, pp. 4849–4859. PMLR, 2020b.

Jin, W., Ma, Y., Liu, X., Tang, X., Wang, S., and Tang, J. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 66–74, 2020c.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in neural information processing systems*, pp. 10215–10224, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Klicpera, J., Bojchevski, A., and Günnemann, S. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations*, 2019.

Kusner, M., Paige, B., and Hernández-Lobato, J. Grammar variational autoencoder. In *Proceedings of the 34 th International Conference on Machine Learning*, Sydney, Australia, PMLR 70, 2017, volume 70, pp. 1945–1954. ACM, 2017.

Landrum, G. et al. RDKit: Open-source cheminformatics. 2006.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Li, Y., Vinyals, O., Dyer, C., Pascanu, R., and Battaglia, P. Learning deep generative models of graphs. In *International Conference on Machine Learning*, 2018.

Lippe, P. and Gavves, E. Categorical normalizing flows via continuous transformations. *arXiv preprint arXiv:2006.09790*, 2020.

Liu, J., Grathwohl, W., Ba, J., and Swersky, K. Graph generation with energy-based models. *ICML Workshop on Graph Representation Learning and Beyond (GRL+)*, 2020a.

Liu, M., Gao, H., and Ji, S. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 338–348, 2020b.

Liu, M., Wang, Z., and Ji, S. Non-local graph neural networks. *arXiv preprint arXiv:2005.14612*, 2020c.

Liu, Q., Allamanis, M., Brockschmidt, M., and Gaunt, A. Constrained graph variational autoencoders for molecule design. *Advances in neural information processing systems*, 31:7795–7804, 2018.

Ma, T., Chen, J., and Xiao, C. Constrained generation of semantically valid graphs via regularizing variational autoencoders. In *Advances in Neural Information Processing Systems*, pp. 7113–7124, 2018.

Ma, Y., Wang, S., Aggarwal, C. C., and Tang, J. Graph convolutional networks with eigenpooling. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 723–731, 2019.

Madhawa, K., Ishiguro, K., Nakago, K., and Abe, M. Graph-nvp: An invertible flow model for generating molecular graphs. *arXiv preprint arXiv:1905.11600*, 2019.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. In *International Conference on Learning Representations*, 2018.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Monti, F., Boscaiini, D., Masci, J., Rodola, E., Svoboda, J., and Bronstein, M. M. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5115–5124, 2017.

Niu, C., Song, Y., Song, J., Zhao, S., Grover, A., and Ermon, S. Permutation invariant graph generation via score-based generative modeling. *Artificial Intelligence and Statistics*, 2020.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.
Popova, M., Shvets, M., Oliva, J., and Isayev, O. Molecular-rnn: Generating realistic molecular graphs with optimized properties. *arXiv preprint arXiv:1905.13372*, 2019.

Ramakrishnan, R., Dral, P. O., Rupp, M., and Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pp. 1530–1538, 2015.

Rogers, D. and Hahn, M. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., and Welling, M. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pp. 593–607. Springer, 2018.

Shi, C., Xu, M., Zhu, Z., Zhang, W., Zhang, M., and Tang, J. Graphaf: a flow-based autoregressive model for molecular graph generation. In *International Conference on Learning Representations*, 2019.

Simonovsky, M. and Komodakis, N. Graphvae: Towards generation of small graphs using variational autoencoders. In *International Conference on Artificial Neural Networks*, pp. 412–422. Springer, 2018.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pp. 11918–11930, 2019.

Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackerman, Z., et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

Turner, R. CD notes. 2005.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representation*, 2018.

Wang, Z., Liu, M., Luo, Y., Xu, Z., Xie, Y., Wang, L., Cai, L., and Ji, S. Advanced graph and sequence neural networks for molecular property prediction and drug discovery. *arXiv preprint arXiv:2012.01981*, 2020.

Weininger, D. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688, 2011.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying graph convolutional networks. In *International Conference on Machine Learning*, pp. 6861–6871, 2019.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. MoleculeNet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Xie, J., Hu, W., Zhu, S.-C., and Wu, Y. N. Learning sparse frame models for natural image patterns. *International Journal of Computer Vision*, 114(2-3):91–112, 2015.

Xie, J., Lu, Y., Zhu, S.-C., and Wu, Y. A theory of generative convnet. In *International Conference on Machine Learning*, pp. 2635–2644, 2016.

Xie, J., Zhu, S.-C., and Nian Wu, Y. Synthesizing dynamic patterns by spatial-temporal generative convnet. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 7093–7101, 2017.

Xie, J., Zheng, Z., Gao, R., Wang, W., Zhu, S.-C., and Nian Wu, Y. Learning descriptor networks for 3d shape synthesis and analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8629–8638, 2018.

Xie, J., Xu, Y., Zheng, Z., Zhu, S.-C., and Nian Wu, Y. Generative pointnet: Energy-based learning on unordered point sets for 3d generation, reconstruction and classification. *arXiv*, pp. arXiv–2004, 2020.

Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i., and Jegelka, S. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pp. 5449–5458, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., et al. Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 59(8):3370–3388, 2019.

Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., and Leskovec, J. Hierarchical graph representation learning with differentiable pooling. In *Advances in neural information processing systems*, pp. 4800–4810, 2018.
You, J., Liu, B., Ying, Z., Pande, V., and Leskovec, J. Graph convolutional policy network for goal-directed molecular graph generation. In Advances in neural information processing systems, pp. 6410–6421, 2018.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. Advances in Neural Information Processing Systems, 33, 2020.

Yuan, H. and Ji, S. Structpool: Structured graph pooling via conditional random fields. In International Conference on Learning Representations, 2020.

Zang, C. and Wang, F. Moflow: an invertible flow model for generating molecular graphs. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 617–626, 2020.

Zhang, M. and Chen, Y. Link prediction based on graph neural networks. In Advances in Neural Information Processing Systems, pp. 5165–5175, 2018.

Zhang, M., Cui, Z., Neumann, M., and Chen, Y. An end-to-end deep learning architecture for graph classification. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

Zhu, S. C., Wu, Y., and Mumford, D. Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling. International Journal of Computer Vision, 27(2):107–126, 1998.
Supplementary Material

A. Molecule Generation

The generation performance on QM9 is given in Table 3.

Table 3. Generation performance on QM9. The results of CVAE and GVAE are obtained from GraphVAE paper. The result of MoFlow is obtained by evaluating its public trained model. All other results are from their original papers.

| Method   | Validity (%) | Uniqueness (%) | Novelty (%) |
|----------|--------------|----------------|-------------|
| CVAE     | 10.30        | 67.50          | 90.00       |
| GVAE     | 60.20        | 9.30           | 80.90       |
| GraphVAE | 55.70        | 76.00          | 61.60       |
| RVAE     | 96.60        | -              | 97.50       |
| MoGAN    | 98.10        | 10.40          | 94.20       |
| GraphNVP | 83.10±0.50   | 99.20±0.30     | 58.20±1.90  |
| GRF      | 84.50±0.70   | 66.00±1.14     | 58.60±0.82  |
| GraphAF  | 100.00       | 94.51          | 88.83       |
| MoFlow   | 100.00±0.00  | 98.53±0.14     | 96.04±0.10  |
| GraphEBM | 100.00±0.00  | 97.90±0.14     | 97.01±0.17  |

B. Goal-Directed Generation

More examples of molecule optimization towards plogp and QED are available in Figure 6.

Figure 6. More examples of molecule optimization towards plogp and QED. The values above and below arrows denote the similarity scores and improvements, respectively. The modifications are highlighted with red rectangles.