MoFlow: An Invertible Flow Model for Molecular Graph Generation

Chengxi Zang¹ Fei Wang¹

Abstract

Generating molecular graphs with desired chemical properties driven by deep generative models provides a very promising way to accelerate the drug discovery process. Such generative models usually consist of two steps: 1) learning latent representations, and 2) generation of molecular graphs. However, to generate novel and chemically-valid molecular graphs from latent representations is very challenging because of the chemical constraints and combinatorial complexity of molecular graphs. In this paper, we propose MoFlow, a flow-based model to learn invertible mappings between molecular graphs and their latent representations. MoFlow generates molecules by first generating bond skeleton through a Glow based model, then generating atoms given bonds by a novel graph conditional flow, and finally combing them into a molecule with posthoc validity correction. MoFlow has merits including exact and tractable likelihood training, efficient one-pass embedding and generation, chemical validity guarantees, 100% reconstruction of training data, and good generalization ability. We validate our model by four tasks: molecular generation and reconstruction, visualization of the continuous latent space, property optimization, and constrained property optimization. Our MoFlow achieves state-of-the-art performance, which implies its potential efficiency and effectiveness to explore large chemical space for drug discovery.

1. Introduction

Drug discovery aims at finding candidate molecules with desired chemical properties for clinical trials, which is a long (10-20 years) and costly (\$0.5-\$2.6 billion) process with a high failure rate (Paul et al., 2010; Avorn, 2015). Recently, deep generative models have demonstrated their big potential to accelerate the drug discovery process by exploring large chemical space in a data-driven manner (Jin et al., 2018; Zhavoronkov et al., 2019). These models usually first learn a continuous latent space by encoding the training molecules and then generate novel and optimized molecules through decoding from the learned latent space guided by targeted properties (Gómez-Bombarelli et al., 2018; Jin et al., 2018). However, it is still very challenging to generate novel and chemically-valid molecules with desired properties since: a) the scale of the chemical space of drug-like compounds is 10^{60} (Mullard, 2017) but the scale of the possibly decoded novel molecules from the learned latent space for searching is much smaller, and b) decoding molecular graphs with chemical constraints (e.g., valency constraints) is a hard combinatorial task.

Prior works leverage different deep generative frameworks for generating molecular SMILES codes (Weininger et al., 1989) or molecular graphs, including variational autoencoder (VAE)-based models (Kusner et al., 2017; Dai et al., 2018; Simonovsky & Komodakis, 2018; Ma et al., 2018; Liu et al., 2018; Bresson & Laurent, 2019; Jin et al., 2018), generative adversarial networks (GAN)-based models (De Cao & Kipf, 2018; You et al., 2018), and autoregressive (AR)based models (Popova et al., 2019; You et al., 2018). In this paper, we explore a different deep generative framework, namely the normalizing flow (Dinh et al., 2014; Madhawa et al., 2019; Kingma & Dhariwal, 2018), for molecular graph generation. Compared with above three frameworks, the flow-based models are the only one which can memorize and exactly reconstruct all the input data, and at the same time have the potential to generate more novel, unique and valid molecules, which implies its potential capability of deeper exploration of the huge chemical space. To our best knowledge, there have been three flow-based models proposed for molecular graph generation. The GraphAF (Shi et al., 2020) model is an autoregressive flow-based model

^{*}Equal contribution ¹Department of Healthcare Policy and Research, Weill Cornell Medicine, Cornell University, NYC, USA. Correspondence to: Chengxi Zang <chz4001@med.cornell.edu>.

Second workshop on *Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models* (ICML 2020), Virtual Conference

that achieves the state-of-the-art performance in molecule generation. GraphAF generates molecules in a sequential manner with validity check when adding a new atom or bond. GraphNVP (Madhawa et al., 2019) and GRF (Honda et al., 2019) are proposed for molecular graph generation in a one-shot manner. However, they cannot guarantee chemical validity and thus show poor performance in generating valid and novel molecules.

In this paper, we propose a novel model named MoFlow for molecular graph generation. MoFlow is first of its kind which not only generates molecules efficiently by invertible mapping at one shot, but also has a validity guarantee. More specifically, to capture the intrinsic atom-and-bond composition of molecules, we propose a variant of the Glow model (Kingma & Dhariwal, 2018) for bonds generation, a novel graph conditional flow for generating atoms given bonds. Then the atoms and bonds are combined into a molecular graph with post-hoc validity correction. MoFlow is trained by exact and tractable likelihood estimation, and one-pass inference and generation can be efficiently utilized for drug optimization by the pre-trained MoFlow.

We validate MoFlow through a wide range of experiments from molecular generation, reconstruction, visualization to optimization. As baselines, we compare the state-ofthe-art VAE-based model (Jin et al., 2018), autoregressivebased models (You et al., 2018; Popova et al., 2019), and all three flow-based models (Madhawa et al., 2019; Honda et al., 2019; Shi et al., 2020). As for memorizing input data, MoFlow achieves 100% reconstruction rate. As for exploring the unknown chemical space, MoFlow outperforms above models by generating more novel, unique and valid molecules (as demonstrated by the N.U.V. scores in Table 2 and 3). MoFlow generates 100% chemically-valid molecules when sampling from prior distributions. Furthermore, if without validity correction, MoFlow still generates much more valid molecules than existing models (validitywithout-check scores in Table 2 and 3). For example, the state-of-the-art autoregressive-flow-based model GraphAF (Shi et al., 2020) achieves 67% and 68% validity-withoutcheck scores for two datasets while MoFlow achieves 96%and 82% respectively, thanks to its capability of capturing the chemical structures in a holistic way. As for chemical property optimization, MoFlow can find much more novel molecules with top drug-likeness scores than existing models (Table 4 and Figure 5). As for constrained property optimization, MoFlow finds novel and optimized molecules with the best similarity scores and second best property improvement (Table 5).

It is worthwhile to highlight our contributions as follows:

 Novel MoFlow model: MoFlow is one of the first flow-based models which not only generates molecular graphs at one shot by invertible mapping but also has a validity guarantee. To capture the intrinsic atom-andbond composition of molecules, we propose a novel graph conditional flow for atoms and a variant of Glow model for bonds, and then combine them with post-hoc validity correction.

• The state-of-the-art performance: MoFlow achieves many state-of-the-art results w.r.t. molecular generation, reconstruction, optimization, etc., and at the same time our one-shot inference and generation are very efficient, which implies its potentials in deep exploration of the large chemical space for drug discovery.

The outline of this paper is: survey (Sec. 2), proposed method (Sec. 3 and 4), experiments (Sec. 5), and conclusions (Sec. 6).

2. Related Work

Molecular Generation. Different deep generative frameworks are proposed for generating molecular SMILES or molecular graphs. Among the variational autoencoder (VAE)-based models (Kusner et al., 2017; Dai et al., 2018; Simonovsky & Komodakis, 2018; Ma et al., 2018; Liu et al., 2018; Bresson & Laurent, 2019; Jin et al., 2018), the JT-VAE (Jin et al., 2018) generates valid tree-structured molecules by first generating a tree-structured scaffold of chemical substructures and then assembling substructures according to the generated scaffold. The MolGAN (De Cao & Kipf, 2018) is a generative adversarial networks (GAN)-based model but shows very limited performance in generating valid and unique molecules. The autoregressive-based models generate molecules in a sequential manner with validity check at each generation step. For example, the MolecularRNN (Popova et al., 2019) sequentially generates each character of SMILES and the GCPN (You et al., 2018) sequentially generates each atom/bond in a molecular graphs. In this paper, we explore a different deep generative framework, namely the normalizing flow models (Dinh et al., 2014; Madhawa et al., 2019; Kingma & Dhariwal, 2018), for molecular graph generation, which have the potential to memorize and reconstruct all the training data and generalize to generating more valid, novel and unique molecules.

Flow-based Models. The (normalizing) flow-based models try to learn mappings between complex distributions and simple prior distributions through invertible neural networks and such a framework has good merits of exact and tractable likelihood estimation for training, efficient one-pass inference and sampling, invertible mapping and thus reconstructing all the training data etc. Examples include NICE(Dinh et al., 2014), RealNVP(Dinh et al., 2016), Glow(Kingma & Dhariwal, 2018) and GNF (Liu et al., 2019) which show promising results in generating images or even graphs (Liu et al., 2019). See latest reviews in (Papamakarios et al., 2019; Kobyzev et al., 2019) and more technical details in

Section 3.

To our best knowledge, there are three flow-based models for molecular graph generation. The GraphAF (Shi et al., 2020) is an autoregressive flow-based model which achieves the state-of-the-art performance in molecule generation. The GraphAF generates molecules in a sequential manner with validity check when adding a new atom or bond. The GraphNVP (Madhawa et al., 2019) and GRF (Honda et al., 2019) are proposed for molecule generation in a one-shot manner. However, they have no guarantee for chemical validity and thus show very limited performance in generating valid and novel molecules. Our MoFlow is first of its kind which not only generates molecules efficiently by invertible mapping at one shot but also has a validity guarantee. In order to capture the atom-and-bond composition of molecules, we propose a variant of Glow(Kingma & Dhariwal, 2018) model for bonds and a novel graph conditional flow for atoms given bonds, and then combining them with a post-hoc validity correction. Our MoFlow achieves many new state-of-the-art results thanks to capturing the chemical structures in a holistic way, and our one-shot inference and generation are more efficient than sequential ones.

3. Model Preliminary

The flow framework. The flow-based models aim to learn a sequence of invertible transformations $f_{\Theta} = f_L \circ ... \circ f_1$ between complex high-dimensional data $X \sim P_{\mathcal{X}}(X)$ and $Z \sim P_{\mathcal{Z}}(Z)$ in a latent space with the same number of dimensions where the latent distribution $P_{\mathcal{Z}}(Z)$ is easy to model (e.g., strong independence assumptions hold in such a latent space). The potentially complex data in the original space can be modelled by the **change of variable formula** where $Z = f_{\Theta}(X)$ and:

$$P_{\mathcal{X}}(X) = P_{\mathcal{Z}}(Z) \mid \det(\frac{\partial Z}{\partial X}) \mid .$$
 (1)

To sample $\tilde{X} \sim P_{\mathcal{X}}(X)$ is achieved by sampling $\tilde{Z} \sim P_{\mathcal{Z}}(Z)$ and then to transform $\tilde{X} = f_{\Theta}^{-1}(\tilde{Z})$ by the reverse mapping of f_{Θ} .

Let $Z = f_{\Theta}(X) = f_L \circ \dots \circ f_1(X)$, $H_l = f_l(H_{l-1})$ where f_l $(l = 1, \dots L \in \mathbb{N}^+)$ are invertible mappings, $H_0 = X$, $H_L = Z$ and $P_Z(Z)$ follows a standard isotropic Gaussian with independent dimensions. Then we get the log-likelihood of X by the change of variable formula as follows:

$$\log P_{\mathcal{X}}(X) = \log P_{\mathcal{Z}}(Z) + \log |\det(\frac{\partial Z}{\partial X})|$$
$$= \sum_{i} \log P_{\mathcal{Z}_{i}}(Z_{i}) + \sum_{l=1}^{L} \log |\det(\frac{\partial f_{l}}{\partial H_{l-1}})|$$
⁽²⁾

where $P_{Z_i}(Z_i)$ is the probability of the i^{th} dimension of Z and $f_{\Theta} = f_L \circ ... \circ f_1$ is an invertible deep neural network to be learnt. Thus, the exact-likelihood-based training is tractable.

Invertible affine coupling layers. However, how to design a.) an invertible function f_{Θ} with b.) expressive structures and c.) efficient computation of the Jacobian determinant are nontrivial. The NICE(Dinh et al., 2014) and RealNVP (Dinh et al., 2016) design an affine coupling transformation $Z = f_{\Theta}(X) : \mathbb{R}^n \mapsto \mathbb{R}^n$:

$$Z_{1:d} = X_{1:d}$$

$$Z_{d+1:n} = X_{d+1:n} \odot e^{S_{\Theta}(X_{1:d})} + T_{\Theta}(X_{1:d}),$$
(3)

by splitting X into two partitions $X = (X_{1:d}, X_{d+1:n})$. Thus, a.) the invertibility is guaranteed by:

$$X_{1:d} = Z_{1:d}$$

$$X_{d+1:n} = (Z_{d+1:n} - T_{\Theta}(Z_{1:d}))/e^{S_{\Theta}(Z_{1:d})},$$
(4)

b.) the expressive power depends on arbitrary neural structures of the **Scale function** $S_{\Theta} : \mathbb{R}^d \mapsto \mathbb{R}^{n-d}$ and the **Transformation function** $T_{\Theta} : \mathbb{R}^d \mapsto \mathbb{R}^{n-d}$ in the affine transformation of $X_{d+1:n}$, and c.) the Jacobian determiant can be computed efficiently by $\det(\frac{\partial Z}{\partial X}) = \exp(\sum_j S_{\Theta}(X_{1:d})_j)$.

Splitting Dimensions. The flow-based models, e.g., Real-NVP (Dinh et al., 2016) and Glow (Kingma & Dhariwal, 2018), adopt **squeeze operation** which compresses the spatial dimension $X^{c \times n \times n}$ into $X^{(ch^2) \times \frac{n}{h} \times \frac{n}{h}}$ to make more channels and then split channels into two halves for the coupling layer. A deep flow model at a specific layer transforms unchanged dimensions in the previous layer to keep all the dimensions transformed. In order to learn an optimal partition of X, Glow (Kingma & Dhariwal, 2018) model introduces an **invertible** 1×1 **convolution** : $\mathbb{R}^{c \times n \times n} \times \mathbb{R}^{c \times c} \mapsto$ $\mathbb{R}^{c \times n \times n}$ with learnable convolution kernel $W \in \mathbb{R}^{c \times c}$ which is initialized as a random rotation matrix. After the transformation Y = invertible 1×1 convolution(X, W), a fixed partition $Y = (Y_{1:\frac{c}{2},::,:}, Y_{\frac{c}{2}+1:n,:,:})$ over the channel cis used for the affine coupling layers.

Numerical stability by actnorm. In order to ensure the numerical stability of the flow-based models, actnorm layer is introduced in Glow (Kingma & Dhariwal, 2018) which normalizes dimensions in each channel over a batch by an affine transformation with learnable scale and bias. The scale and the bias are initialized as the mean and the inverse of the standard variation of the dimensions in each channel over the batch.

4. Proposed MoFlow Model

In this section, we first define the problem and then introduce our <u>Mo</u>lecular <u>Flow</u> (MoFlow) model in detail. We show the outline of our MoFlow in Figure 1 as a roadmap for this section.



Figure 1. The outline of our MoFlow. A molecule M (e.g. Metformin) is represented by a feature matrix A for atoms and adjacency tensors B for bonds. Inference: the graph conditional flow (GCF) $f_{A|B}$ for atoms (Sec. 4.2) transforms A given B into conditional latent vector $Z_{A|B}$, and the Glow f_B for bonds (Sec. 4.3) transform B into latent vector Z_B . The latent space follows a spherical Gaussian distribution. Generation: the generation process is just the reverse transformations of previous operations, followed by a validity correction (Sec. 4.4) procedure which guarantees the chemical validity. We summarize our MoFlow in Sec. 4.5.

4.1. Problem Definition: Learning a Probability Model of Molecular Graphs

Let $\mathcal{M} = \mathcal{A} \times \mathcal{B} \subset \mathbb{R}^{n \times k} \times \mathbb{R}^{c \times n \times n}$ denote the set of \mathcal{M} olecules which is the Cartesian product of the \mathcal{A} tom set \mathcal{A} with at most $n \in \mathbb{N}^+$ atoms belonging to $k \in \mathbb{N}^+$ atom types and the \mathcal{B} ond set \mathcal{B} with $c \in \mathbb{N}^+$ bond types. A molecule $M = (A, B) \in \mathcal{A} \times \mathcal{B}$ is a pair of an atom matrix $A \in \mathbb{R}^{n \times k}$ and a bond tensor $B \in \mathbb{R}^{c \times n \times n}$. We use one-hot encoding for the empirical molecule data where A(i, k) = 1 represents the atom *i* has atom type *k*, and B(c, i, j) = B(c, j, i) = 1 represents a type *c* bond between atom *i* and atom *j*. Thus, a molecule *M* can be viewed as an undirected graph with multi-type nodes and multi-type edges.

Our primary goal is to learn a molecule generative model $P_{\mathcal{M}}(M)$ which is the probability of sampling any molecule M from $P_{\mathcal{M}}$. In order to capture the intrinsic atom-andbond composition of molecules, we decompose the $P_{\mathcal{M}}(M)$ into two parts:

$$P_{\mathcal{M}}(M) = P_{\mathcal{M}}((A,B)) \approx P_{\mathcal{A}|\mathcal{B}}(A|B;\theta_{\mathcal{A}|\mathcal{B}})P_{\mathcal{B}}(B;\theta_{\mathcal{B}})$$
(5)

where $P_{\mathcal{M}}$ is the distribution of molecule, $P_{\mathcal{B}}$ is the distribution of bonds, and $P_{\mathcal{A}|\mathcal{B}}$ is the conditional distribution of atoms given the bonds. The $\theta_{\mathcal{B}}$ and $\theta_{\mathcal{A}|\mathcal{B}}$ are learnable modelling parameters. In contrast with VAE or GAN based frameworks, we can learn the parameters by exact maximum likelihood estimation (MLE) framework by maximizing:

$$\underset{\theta_{\mathcal{B}},\theta_{\mathcal{A}|\mathcal{B}}}{\operatorname{arg\,max}} \mathbb{E}_{M=(A,B)\sim p_{\mathcal{M}-data}} [\log P_{\mathcal{A}|\mathcal{B}}(A|B;\theta_{\mathcal{A}|\mathcal{B}}) + \log P_{\mathcal{B}}(B;\theta_{\mathcal{A}|\mathcal{B}})]$$
(6)

Our model thus consists of two parts, namely a *graph conditional flow for atoms* to learn the atom matrix conditional on the bond tensors and a *flow for bonds* to learn bond tensors.

4.2. Graph Conditional Flow for Atoms

Given a bond tensor $B \in \mathcal{B} \subset \mathbb{R}^{c \times n \times n}$, our goal of the atom flow is to generate the right atom-type matrix $A \in \mathcal{A} \subset \mathbb{R}^{n \times k}$ to assemble valid molecules $M = (A, B) \in \mathcal{M} \subset \mathbb{R}^{n \times k + c \times n \times n}$. We first define \mathcal{B} -conditional flow and graph conditional flow $f_{\mathcal{A}|\mathcal{B}}$ to transform A given Binto conditional latent variable $Z_{A|B} = f_{\mathcal{A}|\mathcal{B}}(A|B)$ which follows isotropic Gaussian $P_{\mathcal{Z}_{\mathcal{A}|\mathcal{B}}}$. We can get the conditional probability of atom features given the bond graphs $P_{\mathcal{A}|\mathcal{B}}$ by a conditional version of the change of variable formula.

4.2.1. B-Conditional Flow and Graph Conditional Flow

Definition 4.1. *B*-conditional flow: A *B*-conditional flow $Z_{A|B}|B = f_{\mathcal{A}|\mathcal{B}}(A|B)$ is an invertible and dimensionkept mapping and there exists reverse transformation $f_{\mathcal{A}|\mathcal{B}}^{-1}(Z_{A|B}|B) = A|B$ where $f_{\mathcal{A}|\mathcal{B}}$ and $f_{\mathcal{A}|\mathcal{B}}^{-1}: \mathcal{A} \times \mathcal{B} \mapsto \mathcal{A} \times \mathcal{B}$.

The condition $B \in \mathcal{B}$ keeps fixed during the transformation. Under the independent assumption of \mathcal{A} and \mathcal{B} , the Jacobian of $f_{\mathcal{A}|\mathcal{B}}$ is:

$$\frac{\partial f_{\mathcal{A}|\mathcal{B}}}{\partial(A,B)} = \begin{bmatrix} \frac{\partial f_{\mathcal{A}|\mathcal{B}}}{\partial A} & \frac{\partial f_{\mathcal{A}|\mathcal{B}}}{\partial B}\\ 0 & \mathbb{1}_B \end{bmatrix},\tag{7}$$

the determiant of this Jacobian is det $\frac{\partial f_{A|B}}{\partial (A,B)} = \det \frac{\partial f_{A|B}}{\partial A}$, and thus the *conditional version of the change of variable formula* in the form of log-likelihood is:

$$\log P_{\mathcal{A}|\mathcal{B}}(A|B) = \log P_{\mathcal{Z}_{\mathcal{A}}|\mathcal{B}}(Z_{A|B}) + \log \mid \det \frac{\partial f_{\mathcal{A}|\mathcal{B}}}{\partial A} \mid.$$
(8)

Definition 4.2. Graph conditional flow: A graph conditional flow is a \mathcal{B} -conditional flow $Z_{A|B}|B = f_{\mathcal{A}|\mathcal{B}}(A|B)$ where $B \in \mathcal{B} \subset \mathbb{R}^{c \times n \times n}$ is the adjacency tenor for edges with *c* types and $A \in \mathcal{A} \subset \mathbb{R}^{n \times k}$ is the feature matrix of the corresponding *n* nodes.

4.2.2. GRAPH COUPLING LAYER

We construct aforementioned invertible mapping $f_{\mathcal{A}|\mathcal{B}}$ and $f_{\mathcal{A}|\mathcal{B}}^{-1}$ by the scheme of the affine coupling layer. Different from traditional affine coupling layer, our coupling transformation relies on graph convolution (Sun et al., 2019) and thus we name such a coupling transformation as a **graph coupling layer**.

 $[\theta_{\mathcal{B}})]$ For each graph coupling layer, we split input $A \in \mathbb{R}^{n \times k}$ into two parts $A = (A_1, A_2)$ along the *n* row dimension, and



Figure 2. Graph conditional flow $f_{\mathcal{A}|\mathcal{B}}$ for the atom matrix. We show the details of one invertible graph coupling layer and a multiscale structure consists of a cascade of *L* layers of such graph coupling layer. The graphnorm is computed only once.

we get the output $Z_{A|B} = (Z_{A_1|B}, Z_{A_2|B}) = f_{\mathcal{A}|\mathcal{B}}(A|B)$ as follows:

$$Z_{A_1|B} = A_1$$

$$Z_{A_2|B} = A_2 \odot \text{Sigmoid}(S_{\Theta}(A_1|B)) + T_{\Theta}(A_1|B).$$
(9)

We deign the scale function S_{Θ} and the transformation function T_{Θ} in each graph coupling layer by incorporating **graph convolution** structures. The bond tensor $B \in \mathbb{R}^{c \times n \times n}$ keeps a fixed value during transforming the atom matrix A. We also apply the masked convolution idea in (Dinh et al., 2016) to the graph convolution in the graph coupling layer. Here, we adopt Relational Graph Convolutional Networks (R-GCN) (Schlichtkrull et al., 2018) to build graph convolution layer **graphconv** as follows:

$$\operatorname{graphconv}(A_1) = \sum_{i=1}^{c} \hat{B}_i(M \odot A) W_i + (M \odot A) W_0 \quad (10)$$

where $\hat{B}_i = D^{-1}B_i$ is the normalized adjacency matrix at channel $i, D = \sum_{c,i} B_{c,i,j}$ is the sum of the in-degree over all the channels for each node, and $M \in \{0,1\}^{n \times k}$ is a binary mask to select a partition A_1 from A. Because the bond graph is fixed during graph coupling layer and thus the graph normalization, denoted as **graphnorm**, is computed only once.

We use multiple stacked graphconv-¿BatchNorm1d-¿ReLu layers with a multi-layer perceptron (MLP) output layer to build the graph scale function S_{Θ} and the graph transformation function T_{Θ} . What's more, instead of using exponential function for the S_{Θ} as discussed in Sec. 3, we adopt Sigmoid function for the sake of the numerical stability of cascading multiple flow layers. The reverse mapping of the graph coupling layer $f_{\mathcal{A}|\mathcal{B}}^{-1}$ is:

$$A_{1} = Z_{A_{1}|B}$$

$$A_{2} = (Z_{A_{2}|B} - T_{\Theta}(Z_{A_{1}|B}|B)) / \text{Sigmoid}(S_{\Theta}(Z_{A_{1}|B}|B)).$$
(11)

The logarithm of the Jacobian determiant of each graph coupling layer can be efficiently computed by:

$$\log \mid \det(\frac{\partial f_{\mathcal{A}|\mathcal{B}}}{\partial A}) \mid = \sum_{j} \log \operatorname{Sigmoid}(S_{\Theta}(A_1|B))_j. \quad (12)$$

In principle, we can use arbitrary complex graph convolution structures for S_{Θ} and T_{Θ} since the computing of above Jacobian determinant of $f_{\mathcal{A}|\mathcal{B}}$ does not involve in computing the Jacobian of S_{Θ} or T_{Θ} .

4.2.3. ACTNORM FOR 2-DIMENSIONAL MATRIX

For the sake of numerical stability, we design a variant of invertible actnorm layer (Kingma & Dhariwal, 2018) for the 2-dimensional atom matrix, denoted as **actnorm2D** (activation normalization for 2D matrix), to normalize each row, namely the feature dimension for each node, over a batch of 2-dimensional atom matrices. Given the mean $\mu \in \mathbb{R}^{n \times 1}$ and the standard deviation $\sigma^2 \in \mathbb{R}^{n \times 1}$ for each row dimension, the normalized input follows $\hat{A} = \frac{A - \mu}{\sqrt{\sigma^2 + \epsilon}}$ where ϵ is a small constant, the reverse transformation is $A = \hat{A} * \sqrt{\sigma^2 + \epsilon} + \mu$, and the logarithmic Jacobian determiant is:

$$\log |\det \frac{\partial \text{actnorm2D}}{\partial X}| = \frac{k}{2} \sum_{i}^{n} |\log(\sigma_{i}^{2} + \epsilon)| \qquad (13)$$

4.2.4. DEEP ARCHITECTURES

We summarize our deep graph conditional flow in Figure 2. We stack multiple graph coupling layers to form graph conditional flow. We alternate different partition of $A = (A_1, A_2)$ in each layer to transform the unchanged part of the previous layer.

4.3. Glow for Bonds

The bond flow aims to learn an invertible mapping $f_{\mathcal{B}} : \mathcal{B} \subset \mathbb{R}^{c \times n \times n} \mapsto \mathcal{B} \subset \mathbb{R}^{c \times n \times n}$ where the transformed latent variable $Z_B = f_{\mathcal{B}}(B)$ follows isotropic Gaussian. According to the change of variable formula, we can get the logarithmic probability of bonds by $\log P_{\mathcal{B}}(B) = \log P_{\mathcal{Z}_{\mathcal{B}}}(Z_B) + \log |\det(\frac{\partial f_{\mathcal{B}}}{\partial B})|$ and generating bond tensor by reversing the mapping $\tilde{B} = f_{\mathcal{B}}^{-1}(\tilde{Z})$ where $\tilde{Z} \sim P_{\mathcal{Z}}(Z)$. We can use arbitrary flow model for the bond tensor and we build our bond flow $f_{\mathcal{B}}$ based on a variant of Glow (Kingma & Dhariwal, 2018) framework.

We also follow the scheme of affine coupling layer to build invertible mappings. For each affine coupling layer, We split input $B \in \mathbb{R}^{c \times n \times n}$ into two parts $B = (B_1, B_2)$ along the channel *c* dimension, and we get the output $Z_B = (Z_{B_1}, Z_{B_2})$ as follows:

$$Z_{B_1} = B_1$$

$$Z_{B_2} = B_2 \odot \operatorname{Sigmoid}(S_{\Theta}(B_1)) + T_{\Theta}(B_1).$$
(14)



Figure 3. A variant of Glow $f_{\mathcal{B}}$ for bonds' adjacency tensors. And thus the reverse mapping $f_{\mathcal{B}}^{-1}$ is:

$$B_{1} = Z_{B_{1}}$$

$$B_{2} = (Z_{B_{2}} - T_{\Theta}(Z_{B_{1}})) / \text{Sigmoid}(S_{\Theta}(Z_{B_{1}})).$$
(15)

Instead of using exponential function as scale function, we use the Sigmoid function with range (0, 1) to ensure the numerical stability when stacking many layers. We find that exponential scale function leads to a large reconstruction error when the number of affine coupling layers increases. The scale function S_{Θ} and the transformation function T_{Θ} in each affine coupling layer can have arbitrary structures. We use multiple 3×3 conv2d-¿BatchNorm2d-¿ReLu layers to build them. The logarithm of the Jacobian determiant of each affine coupling is

$$\log |\det(\frac{\partial Z_B}{\partial B})| = \sum_{j} \log \operatorname{Sigmoid}(S_{\Theta}(B_1))_j.$$
(16)

In order to learn optimal partition and ensure model's stability and learning rate, we also use the invertible 1×1 convolution layer and actnorm layer adopted in the Glow. In order to get more channels for masking and transformation, we **squeeze** the spatial size of *B* from $\mathbb{R}^{c \times n \times n}$ to $\mathbb{R}^{(c*h*h) \times \frac{n}{h} \times \frac{n}{h}}$ by a factor *h* and apply the affine coupling transformation to the squeezed data. The reverse **unsqueeze** operation is adopted to the output. We summarize our bond flow in Figure 3.

4.4. Validity Correction

Real molecules must follow the valency constraints for each atom, but assembling a molecule from generated bond tensor and atom matrix may lead to chemically invalid ones. Here we define the valency constraint for the i^{th} atom as:

$$\sum_{c,j} c * B(c,i,j) \le \text{Valency}(\text{Atom}_i) + Ch$$
(17)

where $B \in \{0,1\}^{c \times n \times n}$ is the one-hot bond tensor over $c \in \{1,2,3\}$ order of chemical bonds (single, double, triple) and $Ch \in \mathbb{N}$ represents the formal charge. Different from existing valency constraints defined in (You et al., 2018;

Popova et al., 2019), we consider the effect of formal charge which may introduce extra bonds for the charged atoms. For example, ammonium [NH4]⁺ may have 4 bonds for N instead of 3. Similarly, S⁺ and O⁺ may have 3 bonds instead of 2. Here we only consider Ch = 1 for N⁺, S⁺ and O⁺ and make Ch = 0 for other atoms.

In contrast with the existing reject-sampling-based validity check adopted in the autoregressive models (You et al., 2018; Popova et al., 2019), we introduce a new post-hoc validity correction procedure after generating a molecule M at once: 1) check the valency constraints of M; 2) if all the atoms of M follows valecny constraints, we return the largest connected component of the molecule Mand end the procedure; 3) if there exists an invalid atom i, namely $\sum_{c,j} cB(c, i, j) > \text{Valency}(\text{Atom}_i) + Ch$, we sort the bonds of i by their order and delete 1 order for the bond with the largest order; 4) go to step 1). Our validity correction procedure tries to make a minimum change to the existing molecule and to keep the largest connected component as large as possible.

4.5. Inference and Generation

We summarize the inference and generation procedure of our MoFlow in Algorithm 1 and Algorithm 2 respectively. We visualize the overall framework in Figure 1. As shown in the algorithms, our MoFlow have merits of exact likelihood estimation/training, one-pass inference, invertible and onepass generation, and chemical validity guarantee.

Algorithm 1 Exact Likelihood Inference (Encoding) by MoFlow

Input: $f_{A|B}$: graph conditional flow for atoms, f_B : glow for bonds, A: atom matrix, B: bond tensor, P_{Z_*} : isotropic Gaussian distributions. **Output:** Z_M :latent representation for atom M, $\log P_M(M)$: log-

arithmic likelihood of molecule M. $Z_B = f_{\mathcal{B}}(B)$ $\log P_{\mathcal{B}}(B) = \log P_{Z_{\mathcal{B}}}(Z_B) + \log |\det(\frac{\partial f_{\mathcal{B}}}{\partial B})|$ $\hat{B} = \text{graphnorm}(B)$ $Z_{A|B} = f_{\mathcal{A}|\mathcal{B}}(A|\hat{B})$ $\log P_{\mathcal{A}|\mathcal{B}}(A|B) = \log P_{\mathcal{Z}_{\mathcal{A}|\mathcal{B}}}(Z_{A|B}) + \log |\det(\frac{\partial f_{\mathcal{A}|\mathcal{B}}}{\partial A})|$ $Z_M = (Z_{A|B}, Z_B)$ $\log P_{\mathcal{M}}(M) = \log P_{\mathcal{B}}(B) + \log P_{\mathcal{A}|\mathcal{B}}(A|B)$ **Return:** Z_M , $\log P_{\mathcal{M}}(M)$

5. Experiments

Following previous works (Jin et al., 2018; Shi et al., 2020), we validate our MoFlow by answering following questions:

• Molecule Generation and Reconstruction (Sec. 5.1): Can our MoFlow memorize and reconstruct all the training molecule datasets? Can our Algorithm 2 Molecule Generation (Decoding) by the Reverse Transformation of MoFlow

Input: $f_{\mathcal{A}|\mathcal{B}}$: graph conditional flow for atoms, $f_{\mathcal{B}}$: glow for bonds, Z_M :latent representation of molecule M or sampling from a prior Gaussian, validity-correction: validity correction rules. **Output:** M: a molecule

 $(Z_{A|B}, Z_B) = Z_M$ $B = f_B^{-1}(Z_B)$ $\hat{B} = \text{graphnorm}(B)$ $A = f_{A|B}^{-1}(Z_{A|B}|\hat{B})$ M = validity-correction(A, B)**Return:** M discrete one-hot-encoded data by adding uniform random noise U[0, 0.6] for each dimension, leading to atom matrix $A \in \mathbb{R}^{9 \times 5}$ and bond tensor $B \in \mathbb{R}^{4 \times 9 \times 9}$ for QM9, and $A \in \mathbb{R}^{38 \times 10}$ and $B \in \mathbb{R}^{4 \times 38 \times 38}$ for ZINC250k.

	Table 1. St	atistics of th	e datasets.	
	#Mol.	Max.	#Node	#Edge
	Graphs	#Nodes	Types	Types
QM9	133,885	9	4+1	3+1
ZINC250K	249,455	38	9+1	3+1

MoFlow generalize to generate novel, unique and valid molecules as many as possible?

- Visualizing Continuous Latent Space (Sec. 5.2): Can our MoFlow embed molecular graphs into continuous latent space with reasonable chemical similarity?
- Property Optimization (Sec. 5.3): Can our MoFlow generate novel molecules with optimized properties?
- **Constrained Property Optimization (Sec. 5.4):** Can our MoFlow generate novel molecules with the optimized properties and at the same time keep the chemical similarity as much as possible?

Baselines. We compare our MoFlow with: a) the state-ofthe-art VAE-based method JT-VAE (Jin et al., 2018) which captures the chemical validity by encoding and decoding a tree-structured scaffold of molecular graphs; b) the stateof-the-art autoregressive models GCPN (You et al., 2018) and MolecularRNN (MRNN)(Popova et al., 2019) with reinforcement learning for property optimization, which generate molecules in a sequential manner; c) flow-based methods GraphNVP (Madhawa et al., 2019) and GRF (Honda et al., 2019) which generate molecules at one shot and the stateof-the-art autoregressive-flow-based model GraphAF (Shi et al., 2020) which generates molecules in a sequential way.

Datasets. We use two datasets QM9 (Ramakrishnan et al., 2014) and ZINC250K (Irwin et al., 2012) for our experiments and summarize them in Table 1. The QM9 contains 133, 885 molecules with maximum 9 atoms in 4 different types, and the ZINC250K has 249, 455 drug-like molecules with maximum 38 atoms in 9 different types. The molecules are kekulized by the chemical software RDKit (Landrum et al., 2006) and the hydrogen atoms are removed. There are three types of edges, namely single, double, and triple bonds, for all molecules. Following the pre-processing procedure in (Madhawa et al., 2019), we encode each atom and bond by one-hot encoding, pad the molecules which have less than the maximum number of atoms with an virtual atom, augment the adjacency tensor of each molecule by a virtual edge channel representing no bonds between atoms, and dequantize (Madhawa et al., 2019; Dinh et al., 2016) the

MoFlow Setup. To be comparable with one-shot-flow baseline GraphNVP (Madhawa et al., 2019), for the ZINC250K, we adopt 10 coupling layers and 38 graph coupling layers for the bonds' Glow and the atoms' graph conditional flow respectively. We use two 3 * 3 convolution layers with 512,512 hidden dimensions in each coupling layer. For each graph coupling layer, we set one relational graph convolution layer with 256 dimensions followed by a two-layer multilayer perceptron with 512, 64 hidden dimensions. As for the QM9, we adopt 10 coupling layers and 27 graph coupling layers for the bonds' Glow and the atoms' graph conditional flow respectively. There are two 3*3 convolution layers with 128, 128 hidden dimensions in each coupling layer, and one graph convolution layer with 64 dimensions followed by a two-layer multilayer perceptron with 128,64 hidden dimensions in each graph coupling layer. As for the optimization experiments, we further train a regression model to map the latent embeddings to different property scalars (discussed in Sec. 5.3 and 5.4) by a multi-layer perceptron with 18-dim linear layer -¿ ReLu -¿ 1-dim linear layer structures. For each dataset, we use the same trained model for all the following experiments.

Empirical Running Time. Following above setup, we implemented our MoFlow by Pytorch-1.3.1 and trained it by Adam optimizer (Kingma & Ba, 2014) with learning rate 0.001, batch size 256, and 200 epochs for both datasets on 1 GeForce RTX 2080 Ti GPU and 16 CPU cores. Our MoFlow finished 200-epoch training within 22 hours (6.6 minutes/epoch) for ZINC250K and 3.3 hours (0.99 minutes/epoch) for QM9. Thanks to efficient one-pass inference/embedding, our MoFlow takes negligible 7 minutes to learn an additional regression layer trained in 3 epochs for optimization experiments on ZINC250K. In comparison, as for the ZINC250K dataset, GraphNVP (Madhawa et al., 2019) costs 38.4 hours (11.5 minutes/epoch) by our Pytorch implementation for training on ZINC250K with the same configurations, and the estimated total running time of GraphAF (Shi et al., 2020) is 124 hours (24 minutes/epoch) which consists of the reported 4 hours for a generation model trained by 10 epochs and estimated 120 hours for another optimization model trained by 300 epochs. The reported running time of JT-VAE (Jin et al., 2018) is roughly 24 hours in (You et al., 2018).

5.1. Molecule Generation and Reconstruction

Setup. In this task, we evaluate our MoFlow 's capability of generating novel, unique and valid molecules, and if our MoFlow can reconstruct input molecules from their latent representations. We adopted the widely-used metrics, including: Validity which is the percentage of chemically valid molecules in all the generated molecules, Uniqueness which is the percentage of unique valid molecules in all the generated molecules, Novelty which is the percentage of generated valid molecules which are not in the training dataset, and **Reconstruction** rate which is the percentage of molecules in the input dataset which can be reconstructed from their latent representations. Besides, because the novelty score also accounts for the potentially duplicated novel molecules, we propose a new metric N.U.V. which is the percentage of Novel, Unique, and Valid molecules in all the generated molecules. We also compare the validity of ablation models if not using validity check or validity correction, denoted as Validity w/o check in (Shi et al., 2020).

The prior distribution of latent space follows a spherical multivariate Gaussian distribution $\mathcal{N}(0, (t\sigma)^2 \mathbf{I})$ where σ is the learned standard deviation and the hyper-parameter tis the temperature for the reduced-temperature generative model (Parmar et al., 2018; Kingma & Dhariwal, 2018; Madhawa et al., 2019). We use t = 0.85 in the generation for both QM9 and ZINC250K datasets, and t = 0.6for the ablation study without validity correction. To be comparable with the state-of-the-art baseline GraphAF(Shi et al., 2020), we generate 10,000 molecules, i.e., sampling 10,000 latent vectors from the prior and then decode them by the reverse transformation of our MoFlow. We report the the mean and standard deviation of results over 5 runs. As for the reconstruction, we encode all the molecules from the training dataset into latent vectors by the encoding transformation of our MoFlow and then reconstruct input molecules from these latent vectors by the reverse transformation of MoFlow.

Results. Table 2 and Table 3 show that our MoFlow outperfoms the state-of-the-art models on all the six metrics for both QM9 and ZINC250k datasets. Thanks to the invertible characteristic of the flow-based models, our MoFlow builds an one-to-one mapping from the input molecule M to its corresponding latent vector Z, enabling 100% reconstruction rate as shown in Table 2 and Table 3. In contrast, the VAEbased method JT-VAE and the autoregressive-based method GCPN and MRNN can't reconstruct all the input molecules. Compared with the one-shot flow-based model GraphNVP and GRF, by incorporating validity correction mechanism, our MoFlow achieves 100% validity, leading to significant improvements of the validity score and N.U.V. score for both datasets. Specifically, the N.U.V. score of MoFlow are 2 and 3 times as large as the N.U.V. scores of GraphNVP and GRF respectively in Table 2. Even without validity correction, our MoFlow still outperforms the validity scores of GraphNVP and GRF by a large margin. Compared with the autoregressive flow-based model GraphAF, we find our MoFlow outperforms GraphAF by additional 16% and 0.8% with respect to N.U.V scores for QM9 and ZINC respectively, indicating that our MoFlow generates more novel, unique and valid molecules. Indeed, MoFlow achieves better uniqueness score and novelty score compared with GraphAF for both datasets. What's more, our MoFlow without validity correction still outperforms GraphAF without the validity check by a large margin w.r.t. the validity score (validity w/o check in Table 2 and Table 3) for both datasets, implying the superiority of capturing the molecular structures in a holistic way by our MoFlow over autoregressive ones in a sequential way.

In conclusion, our MoFlow not only memorizes and reconstructs all the training molecules, but also generates more novel, unique and valid molecules than existing models, indicating that our MoFlow learns a strict superset of the training data and explores the unknown chemical space better.

5.2. Visualizing Continuous Latent Space

Setup. We examine the learned latent space of our MoFlow , denoted as f, by visualizing the decoded molecules from a neighborhood of a latent vector in the latent space. Similar to (Kusner et al., 2017; Jin et al., 2018), we encode a seed molecule M into Z = f(M) and then grid search two random orthogonal directions with unit vector X and Y based on Z, then we get new latent vector by $Z' = Z + \lambda_X * X + \lambda_Y * Y$ where λ_X and λ_Y are the searching steps. Different from VAE-based models, our MoFlow gets decoded molecules efficiently by the one-pass inverse transformation $M' = f^{-1}(Z')$. In contrast, the VAE-based models such as JT-VAE need to decode each latent vectors 10 - 100 times and autoregressive-based models like GCPN, MRNN and GraphAF need to generate a molecule sequentially. Further more, we measure the chemical similarity between each neighboring molecule and the centering molecule. We choose Tanimoto index (Bajusz et al., 2015) as the chemical similarity metrics and indicate their similarity values by a heatmap. We further visualize a linear interpolation between two molecules to show their changing trajectory in the latent space similar to the interpolation case between images (Kingma & Dhariwal, 2018).

Results. We show the visualization of latent space in Figure 4. We find the latent space is very smooth and the inter-

MoFlow: An Invertible	e Flow Model for	Molecular Gra	ph Generation
-----------------------	------------------	---------------	---------------

			1	· ·		
	% Validity	% Validity w/o check	% Uniqueness	% Novelty	% N.U.V.	% Reconstruct
GraphNVP (Madhawa et al., 2019)	83.1 ± 0.5	n/a	99.2 ± 0.3	58.2 ± 1.9	47.97	100
GRF (Honda et al., 2019)	84.5 ± 0.70	n/a	66.0 ± 1.15	58.6 ± 0.82	32.68	100
GraphAF (Shi et al., 2020)	100	67	94.51	88.83	83.95	100
MoFlow	100.00 ± 0.00	96.17 ± 0.18	99.20 ± 0.12	98.03 ± 0.14	97.24 ± 0.21	100.00 ± 0.00

Table 2. Generation and reconstruction performance on QM9 dataset.

Table 3. Generation and reconstruction performance on ZINC250K dataset.

	% Validity	% Validity w/o check	% Uniqueness	% Novelty	% N.U.V.	% Reconstruct
JT-VAE (Jin et al., 2018)	100	n/a	100	100	100	76.7
GCPN (You et al., 2018)	100	20	99.97	100	99.97	n/a
MRNN (Popova et al., 2019)	100	65	99.89	100	99.89	n/a
GraphNVP (Madhawa et al., 2019)	42.6 ± 1.6	n/a	94.8 ± 0.6	100	40.38	100
GRF (Honda et al., 2019)	73.4 ± 0.62	n/a	53.7 ± 2.13	100	39.42	100
GraphAF (Shi et al., 2020)	100	68	99.10	100	99.10	100
MoFlow	$\underline{100.00}\pm0.00$	81.76 ± 0.21	99.99 ± 0.01	$\underline{100.00}\pm0.00$	99.99 ± 0.01	100.00 ± 0.00

pro Si	gro.	prof	prof	xx Be	Day of	A A	W AS	prof		
0.42	0.41	0.38	0.32	0.44	0.37	0.37	0.37	0.31		- 1.00
pro.	prox	prox	pege	pege	pro p	D and	mag.	prog		
0.53	0.53	0.53	0.68	0.68	0.57	0.57	0.43	0.32		
prof	ond	pige	pro.	prof	pege	prof	prof.	2 gra		- 0.80
1.00	0.52	0.68	0.68	0.68	0.68	0.57	0.47	0.39		
prog	prog	pro	pige	perg	per	pro	pro	pro		0.60 🚽
1.00	1.00	1.00	0.68	0.68	0.68	1.00	1.00	0.57		anir
prof	prof	pro	prog	prog	prog	prog	pro	prof		noto S
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.68		<u>n</u>
301	201	gor	300	901	901	prog	prof	prof		- 0.40 arity
0.59	0.59	0.59	0.59	0.59	0.59	0.79	0.79	0.53		
901	301	201	301	901	901	90e	prof	prof		- 0.20
0.59	0.59	0.59	0.59	0.59	0.59	0.59	0.53	0.71		
9 0	300	300	300	300	300	300	900	And		
0.34	0.59	0.59	0.59	0.59	0.59	0.59	0.54	0.71		0.00
	\$ 0	301	301	500	901 050	901	302	901		
0.32	0.34	0.59	0.59	0.59	0.59	0.32	0.54	0.54		
diro.	Yuo	yahd	, and	des.	67	$\sim c \gamma$	6 d>	~ ~ ~	≻ ਠਿੱ	\sim
1.00	0.71	0.51	0.49	0.38	0.13	0.13	0.14	4 0.19		0.17

Figure 4. Visualization of learned latent space by our MoFlow. Top: Visualization of the grid neighbors of a seed molecule in the center, which serves as the baseline for measuring similarity. Bottom: Interpolation between two seed molecules and the left one is the baseline molecule for measuring similarity. Seed molecules are highlighted in red boxs and they are randomly selected from ZINC250K.

polations between two latent points only change a molecule graph a little bit. Quantitatively, we find the chemical similarity between molecules majorly correspond to their Euclidean distance between their latent vectors, implying that our MoFlow embeds similar molecular structures into similar latent embeddings. Searching in such a continuous latent space learnt by our MoFlow is the basis for molecular property optimization and constraint optimization as discussed in the following sections.

5.3. Property Optimization

Setup. The property optimization task aims at generating novel molecules with the best Quantitative Estimate of Druglikeness (QED) scores (Bickerton et al., 2012) which measures the drug-likeness of generated molecules. Following the previous works (You et al., 2018; Popova et al., 2019), we report the best property scores of novel molecules discovered by each method.

We use the pre-trained MoFlow, denoted as f, in the generation experiment to encode a molecule M and get the molecular embedding Z = f(M), and further train a multilayer perceptron to regress the embedding Z of the molecules to their property values y. We then search the best molecules by the gradient ascend method, namely $Z' = Z + \lambda * \frac{dy}{dZ}$ where the λ is the length of the search step. We conduct above gradient ascend method by K steps. We decode the new embedding Z' in the latent space to the discovered molecule by reverse mapping $M' = f^{-1}(Z')$. The molecule M' is novel if M' doesn't exist in the training dataset.

Table 4. Discovered novel molecules with the best QED scores. Our MoFlow finds more molecules with the best QED scores. More results in Figure 5.

Method	1st	2nd	3rd	4th
ZINC (Dataset)	0.948	0.948	0.948	0.948
JT-VAE	0.925	0.911	0.910	-
GCPN	0.948	0.947	0.946	-
MRNN	0.948	0.948	0.947	-
GraphAF	0.948	0.948	0.947	0.946
MoFlow	0.948	0.948	0.98	0.948



Figure 5. Illustration of discovered novel molecules with the best druglikeness QED scores.

Results. We report the discovered novel molecules sorted

Table 5. Constrained optimization on Penalized-logP

		JT-VAE			GCPN	
δ	Improvement	Similarity	Success	Improvement	Similarity	Success
0.0 0.2 0.4 0.6	$\begin{array}{c} 1.91 \pm 2.04 \\ 1.68 \pm 1.85 \\ 0.84 \pm 1.45 \\ 0.21 \pm 0.71 \end{array}$	$\begin{array}{c} 0.28 \pm 0.15 \\ 0.33 \pm 0.13 \\ 0.51 \pm 0.10 \\ 0.69 \pm 0.06 \end{array}$	97.5% 97.1% 83.6% 46.4%	$\begin{array}{c} 4.20 \pm 1.28 \\ 4.12 \pm 1.19 \\ 2.49 \pm 1.30 \\ 0.79 \pm 0.63 \end{array}$	$\begin{array}{c} {\bf 0.32 \pm 0.12} \\ {0.34 \pm 0.11} \\ {0.48 \pm 0.08} \\ {0.68 \pm 0.08} \end{array}$	100% 100% 100% 100%
		GraphAF			MoFlow	
δ	Improvement	GraphAF Similarity	Success	Improvement	MoFlow Similarity	Success

by their QED scores in Table 4. We find previous methods can only find very few molecules with the best QED score (= 0.948). In contrast, our MoFlow finds much more novel molecules which have the best QED values than all the baselines. We show more molecular structures with top QED values in Figure 5.

5.4. Constrained Property Optimization

Setup. The constrained property optimization aims at finding a new molecule M' with the largest similarity score sim(M, M') and the largest improvement of a targeted property value y(M') - y(M) given a molecule M. Following the similar experimental setup of (Jin et al., 2018; You et al., 2018), we choose Tanimoto similarity of Morgan fingerprint (Rogers & Hahn, 2010) as the similarity metrics, the penalized logP (plogp) as the target property, and Mfrom the 800 molecules with the lowest plogp scores in the training dataset of ZINC250K. We use similar gradient ascend method as discussed in the previous subsetion to search for optimized molecules. An optimization succeeds if we find a novel molecule M' which is different from Mand $y(M') - y(M) \ge 0$ and $sim(M, M') \ge \delta$ within K steps where δ is the smallest similarity threshold to screen the optimized molecules.

Results. We summarize the results in Table 5. We find our MoFlow can find the most similar new molecules at the same time achieve very good plogp improvement. Compared with the state-of-the-art VAE model JT-VAE, our MoFlow achieves much higher similarity score and property improvement, implying that our model is good at interpolation and learning continuous molecular embedding. Compared with the state-of-the-art reinforcement learning based method GCPN and GraphAF which is good at generating molecules step-by-step with targeted property rewards, our model MoFlow achieves the best similarity scores and the second best property improvements. We illustrate one optimization example in Figure 6 with very similar structures but a large improvement w.r.t the penalized logP.



Figure 6. An illustration of the constrained optimization of a molecule leading to an improvement of +16.48 w.r.t the penalized logP and with Tanimoto similarity 0.624. The modified part is highlighted.

6. Conclusion

In this paper, we propose a novel model named MoFlow for molecular graph generation. Our MoFlow is one of the first flow-based models which not only generates molecular graphs at one-shot by invertible mappings but also has a validity guarantee. Our MoFlow consists of a variant of Glow model for bonds, a novel graph conditional flow for atoms given bonds, and then combining them with posthoc validity correction. Our MoFlow achieves many new state-of-the-art performance on molecular generation, reconstruction and optimization. For future work, we try to combine the advantages of both sequential generative models and one-shot generative models to generate chemically feasible molecular graphs.

References

- Avorn, J. The \$2.6 billion pill—methodologic and policy considerations. *New England Journal of Medicine*, 372 (20):1877–1879, 2015.
- Bajusz, D., Rácz, A., and Héberger, K. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of cheminformatics*, 7(1):20, 2015.
- 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, 2012.
- Bresson, X. and Laurent, T. A two-step graph convolutional decoder for molecule generation. *arXiv preprint arXiv:1906.03412*, 2019.
- Dai, H., Tian, Y., Dai, B., Skiena, S., and Song, L. Syntaxdirected variational autoencoder for structured data. arXiv preprint arXiv:1802.08786, 2018.
- 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.

- 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.
- Honda, S., Akita, H., Ishiguro, K., Nakanishi, T., and Oono, K. Graph residual flow for molecular graph generation. *arXiv preprint arXiv:1909.13521*, 2019.
- 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.
- Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. arXiv preprint arXiv:1802.04364, 2018.
- Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
- Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In Advances in Neural Information Processing Systems, pp. 10215–10224, 2018.
- Kobyzev, I., Prince, S., and Brubaker, M. A. Normalizing flows: Introduction and ideas. *arXiv preprint arXiv:1908.09257*, 2019.
- Kusner, M. J., Paige, B., and Hernández-Lobato, J. M. Grammar variational autoencoder. In *Proceedings of* the 34th International Conference on Machine Learning-Volume 70, pp. 1945–1954. JMLR. org, 2017.
- Landrum, G. et al. Rdkit: Open-source cheminformatics, 2006.
- Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), Proceedings of the 17th International Conference on Machine Learning (ICML 2000), pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.
- Liu, J., Kumar, A., Ba, J., Kiros, J., and Swersky, K. Graph normalizing flows. In *Advances in Neural Information Processing Systems*, pp. 13556–13566, 2019.
- Liu, Q., Allamanis, M., Brockschmidt, M., and Gaunt, A. Constrained graph variational autoencoders for molecule design. In *Advances in Neural Information Processing Systems*, pp. 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.

- Madhawa, K., Ishiguro, K., Nakago, K., and Abe, M. Graphnvp: An invertible flow model for generating molecular graphs. arXiv preprint arXiv:1905.11600, 2019.
- Mullard, A. The drug-maker's guide to the galaxy. *Nature News*, 549(7673):445, 2017.
- Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.
- Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, Ł., Shazeer, N., Ku, A., and Tran, D. Image transformer. arXiv preprint arXiv:1802.05751, 2018.
- Paul, S. M., Mytelka, D. S., Dunwiddie, C. T., Persinger, C. C., Munos, B. H., Lindborg, S. R., and Schacht, A. L. How to improve r&d productivity: the pharmaceutical industry's grand challenge. *Nature reviews Drug discovery*, 9(3):203, 2010.
- Popova, M., Shvets, M., Oliva, J., and Isayev, O. Molecularrnn: 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*, 1:140022, 2014.
- 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. *ICLR 2020, Addis Ababa, Ethiopia, Apr:26-Apr. 30, 2020, 2020.*
- 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.
- Sun, M., Zhao, S., Gilvary, C., Elemento, O., Zhou, J., and Wang, F. Graph convolutional networks for computational drug development and discovery. *Briefings in bioinformatics*, 2019.
- Weininger, D., Weininger, A., and Weininger, J. L. Smiles. 2. algorithm for generation of unique smiles notation. *Journal of chemical information and computer sciences*, 29(2):97–101, 1989.

- 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.
- Zhavoronkov, A., Ivanenkov, Y. A., Aliper, A., Veselov, M. S., Aladinskiy, V. A., Aladinskaya, A. V., Terentiev, V. A., Polykovskiy, D. A., Kuznetsov, M. D., et al. Deep learning enables rapid identification of potent ddr1 kinase inhibitors. *Nature biotechnology*, 37(9):1038–1040, 2019.