3DLinker: An E(3) Equivariant Variational Autoencoder for Molecular Linker Design

Yinan Huang [1] Xingang Peng [2] Jianzhu Ma [3 1] Muhan Zhang [3 1]

Abstract

Deep learning has achieved tremendous success in designing novel chemical compounds with desirable pharmaceutical properties. In this work, we focus on a new type of drug design problem—generating a small "linker" to physically attach two independent molecules with their distinct functions. The main computational challenges include: 1) the generation of linkers is *conditional* on the two given molecules, in contrast to generating complete molecules from scratch in previous works; 2) linkers heavily depend on the anchor atoms of the two molecules to be connected, which are not known beforehand; 3) 3D structures and orientations of the molecules need to be considered to avoid atom clashes, for which equivariance to E(3) group are necessary. To address these problems, we propose a conditional generative model, named 3DLinker, which is able to predict anchor atoms and jointly generate linker graphs and their 3D structures based on an E(3) equivariant graph variational autoencoder. So far as we know, no previous models could achieve this task. We compare our model with multiple conditional generative models modified from other molecular design tasks and find that our model has a significantly higher rate in recovering molecular graphs, and more importantly, accurately predicting the 3D coordinates of all the atoms.

1. Introduction

The biological functions of most small molecule drugs are to inhibit the activity of the target protein by binding its active sites. In drug discovery, designing new molecule drugs with desired pharmacophoric properties remains challenging due to the discreteness and enormity of the search space (Polishchuk et al., 2013). To address this problem, many machine learning methods have been developed to embed molecules in a compact hidden space, making promising progress in multiple downstream computational tasks such as molecular de-novo design, molecular optimization, and chemical property prediction.

Molecules are generally represented by graphs with atoms and bonds represented as nodes and edges, respectively. Graph generative models (Liu et al., 2018; Shi et al., 2019; Jin et al., 2018; 2020) are commonly applied to model the marginal probability for FDA-approved drug molecules and it is expected that the newly sampled molecules from the model have similar or better pharmacophoric properties.

However, in complex diseases such as cancer, mutations of amino acids could significantly impact the binding affinity between drugs and target proteins. The drug might fall off the drug target when a particular amino acid mutates with a certain probability due to the weak binding affinity, making the patient drug-resistant. To solve this problem, more recently, an alternative drug mechanism named Proteolysis targeting chimera (PROTAC) is developed to inhibit the protein functions by prompting complete degradation of the target protein. PROTAC is a unique molecule composed of two *fragment* molecules and a *linker* molecule: one fragment binds the target protein, the other binds another molecule that can degrade the target protein, and the linker attaches the two fragments together. Because PROTAC needs only to bind their targets with high selectivity (rather than inhibit the target protein's activity), many efforts are devoted to retooling previously ineffective inhibitor molecules as PROTAC for developing the next-generation drugs. Even though PROTAC owns promising potential, it has not been broadly pushed into clinical trial stages. One of the key challenges is the design of linker, which has a critical influence on the ultimate degradation of the target protein. To date, linker design still relies on the expertise of structural biologists and thus is very time-intensive. Therefore, there are increasing efforts to develop deep learning methods to address linker design problems (Imrie et al., 2020; Yang et al., 2020).

[1]Beijing Institute for General Artificial Intelligence [2]Tsinghua University [3]Institute for Artificial Intelligence, Peking University. Correspondence to: Muhan Zhang <muhan@pku.edu.cn>, Jianzhu Ma <majianzhu@pku.edu.cn>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

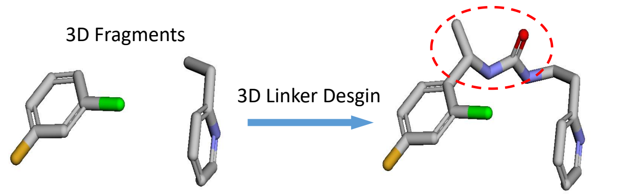

Figure 1: 3D linker design problem: given two fragments' graph with 3D coordinates (left), the goal is to generate a linker graph with 3D coordinates to link these two fragments (right). The 3D coordinates of the generated linker must align with the two fragments, otherwise they cannot link.

A critical challenge for computational linker design stems from its strong 3D spatial constraints compared to classical graph generation tasks. It is known that a successful fragment linker should not disturb the spatial configurations of the two fragments (Ichihara et al., 2011; Klon, 2015). In addition, the anchors between fragments and linker also have correlations with their spatial poses. The linker design problem should be extended to include the 3D information, and a 3D-aware generative model is needed to generate realistic linkers, rather than invoking a graph generative model. In this paper, we propose a conditional generative model, named 3DLinker, that jointly models the 2D molecular graphs and 3D structures of linker for solving the 3D linker design problem.

Given the graph and spatial coordinates of two fragments, 3DLinker can jointly generate graphs and spatial coordinates of the linker. Notably, it does not rely on pre-determined anchors and can accurately predict anchors based on the two observed fragments to be connected. More importantly, 3DLinker can predict 3D coordinates directly and at the same time keeps equivariant to rotations, translations and reflections, which makes it insensitive to choices of the coordinate system. Finally, since the generative model is based on the variational autoencoder (VAE) framework (Kingma & Welling, 2013), it can be used as an unsupervised representation learning method whose latent representations are fed into downstream tasks such as drug-likeness prediction. To the best of our knowledge, 3DLinker is the first trial that simultaneously predicts equivariant graph and 3D coordinates for the linker design problem.

2. Background

In this section we introduce the definition of the 3D linker design problem and basic concepts of E(3) equivariance.

2.1. 3D Linker Design

Linker design is to generate a small molecule that can link two given molecular fragments at specific anchors (binding atoms). Instead of modeling it as a 2D graph generation problem, it is crucial to take 3D information into account, since the designed fragment should satisfy spatial constraints such as not disturbing the relative poses or causing any atom clashes between the two fragments. Therefore, it requires the linker design algorithm to be able to generate both the chemical graph and 3D coordinates given the two fragments and their spatial coordinates (Figure 1).

Mathematically, a molecule can be viewed as a graph with atoms as nodes and chemical bonds as edges. A 3D molecule can be represented by a graph $G = (\mathcal{V}, \mathcal{E}, X)$ with 3D coordinates $\boldsymbol{r} = (x, y, z)$, where $\mathcal{V}$ is the set of nodes, $|\mathcal{V}|$ is the number of nodes, $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ are edges, X are node types and $\boldsymbol{R}$ is a matrix whose i-th row is $\boldsymbol{r}_i^\top$ ($\top$ stands for transpose). In the 3D linker design, two fragments are defined by two unlinked subgraphs $G_\mathrm{F} = (G_{\mathrm{F},1}, G_{\mathrm{F},2})$ with geometry $\boldsymbol{R}_\mathrm{F}$, and a linker is denoted by G_L with geometry $\boldsymbol{R}_\mathrm{L}$. Let $G, \boldsymbol{R}$ be the graph and geometry of the ground truth linked molecule containing both fragments and linker. A 3D linker design model is a conditional generative model that completes the ground truth molecule graph as well as its geometry given the two fragments:

$$p(G, \boldsymbol{R} | G_\mathrm{F}, \boldsymbol{R}_\mathrm{F}). \tag{1}$$

2.2. O(3), E(3) Groups and Equivariance

Group is a set of operations equipped with multiplications, associativity, the identity element and the inverse element. Group of all 3D rotations and reflections is called 3D Orthogonal Group or O(3), and group of all 3D rotations, translations and reflections is called 3D Euclidean Group or E(3). Let $\mathcal{X}$ be the input space, $\mathcal{Y}$ be output space and GL($\mathcal{X}$) be all invertible linear transformations from $\mathcal{X}$ to $\mathcal{X}$ (similar for GL($\mathcal{Y}$)). A function $\phi : \mathcal{X} \to \mathcal{Y}$ is called *equivariant* to the group $\mathcal{G}$, if for all group element $g \in \mathcal{G}$ and all $x \in \mathcal{X}$, there exists group representations $\pi^\mathcal{X} : \mathcal{G} \to$ GL($\mathcal{X}$) and $\pi^\mathcal{Y} : \mathcal{G} \to$ GL($\mathcal{Y}$) such that

$$\pi^\mathcal{Y}(g)\phi(x) = \phi(\pi^\mathcal{X}(g)x). \tag{2}$$

If $\pi^\mathcal{Y}(g)$ is the identity function for all $g \in \mathcal{G}$, then we say ϕ is *invariant* to group $\mathcal{G}$. In 3D linker design, it is known that a molecule graph G should not depend on a specific coordinate system and $\boldsymbol{R}$ should change equivariantly to transformations of the coordinate system. Therefore it raises a constraint that for any $g \in E(3)$, the generative model $p(G, \boldsymbol{R} | G_\mathrm{F}, \boldsymbol{R}_\mathrm{F})$ should satisfy

$$p(G, \pi(g)\boldsymbol{R} | G_\mathrm{F}, \pi(g)\boldsymbol{R}_\mathrm{F}) = p(G, \boldsymbol{R} | G_\mathrm{F}, \boldsymbol{R}_\mathrm{F}), \tag{3}$$

where $\pi(g)$ can by any rotation, translations or reflections matrix in the 3D space.

3. Related Work

Graph-based Molecular Generation. Variational Graph Auto-Encoders are the most popular models for molecular generations. Early approaches mainly focus on embedding the 2D chemical graphs into low dimensional space and sample new molecules by perturbing the hidden values. The representative works include GraphVAE (Simonovsky & Komodakis, 2018), CGVAE (Liu et al., 2018), JT-VAE (Jin et al., 2018), GraphNVP (Madhawa et al., 2019) and so on. Although none of these methods designs molecular linkers, graph-based VAE models serve as the basic building block of our entire architecture. All these models could be plugged into our framework by transforming the generative model into a conditional one. From the perspective of training techniques, auto-regression is also widely adopted to train graph-based deep learning models, such as GraphRNN (You et al., 2018), DeepGMG (Li et al., 2018), GraphAF (Shi et al., 2019). Most of these models generate nodes and edges in a sequential manner.

Point-cloud-based Molecular Models. An important component of our work is to model and design the 3D structures of molecular fragments, in which maintaining the equivariant properties is the crucial computational challenge. One typical solution is to model the molecules as 3D point clouds using equivariant neural networks (Schütt et al., 2017; Klicpera et al., 2020; Liu et al., 2021; Satorras et al., 2021b; Thomas et al., 2018; Fuchs et al., 2020; Deng et al., 2021; Jing et al., 2020). To train such models, auto-regression is a more common solution, such as G-SchNet (Gebauer et al., 2019), G-SphereNet (Anonymous, 2022), and (Luo et al., 2021). But flow-based model ENF (Satorras et al., 2021a) and reinforcement learning (Simm et al., 2020a;b) could also be applied. The main limitation of point-cloud-based models is that they cannot directly generate discrete graph structures, which makes it difficult to model chemical constraints like valency (maximal number of hydrogen atoms one can combine with).

Molecular Linker Design. DeLinker (Imrie et al., 2020) is the first attempt to apply deep learning methods to the linker design problem. It constructs a conditional graph generative model that generates linker given two fragments. It adapts CGAVE (Liu et al., 2018), generating edges step by step starting with fragments and two known anchor nodes as the binding sites. The spatial distance and angle between two fragments are provided to the model as side information to guide the generation. DEVELOP (Imrie et al., 2021) improves DeLinker by encoding the spatial information of fragments using CNN. SyntaLinker (Yang et al., 2020) is a text-based transformer that directly transforms the SMILES (a text representation of molecular graphs) of input fragments into ground-truth ones. However, none of them have a detailed atom-level description of molecule geometry, which is insufficient to express the fragments' geometry. In addition, anchor nodes are either not considered or assumed to be known in advance, the latter of which is rare in a real-world application. Most importantly, they are only able to generate graph representations of linker without 3D coordinates. Gen3D (Roney et al., 2021), though generating both graphs and coordinates equivariantly, is not designed for linker design.

4. Methodology

In this section, we present our 3DLinker, a conditional VAE-based generative model that generates both invariant graphs and equivariant absolute coordinates of linkers given two 3D fragments.

Notations. Let $G_{\mathrm{F}}, G_{\mathrm{L}}, G$ be graphs of fragments, linker and full molecule (ground truth) respectively, and similarly for coordinates $\boldsymbol{R}_{\mathrm{F}}, \boldsymbol{R}_{\mathrm{L}}, \boldsymbol{R}$ as in section 2.1 . As we complete the full molecule graph step by step, we use G_t and $\boldsymbol{R}_t$ to denote the current (existing) graph and coordinates at timestamp t, where $G_0 = G_{\mathrm{F}}, \boldsymbol{R}_0 = \boldsymbol{R}_{\mathrm{F}}$. The encoder embeds each node $i \in \mathcal{V}$ with both invariant features $h_i \in \mathbb{R}^{n_h}$ (for embedding the graph) and equivariant features $\boldsymbol{v}_i \in \mathbb{R}^{n_v \times 3}$ (for embedding the coordinates), which are further used for sampling invariant latent variables $z_i^h \in \mathbb{R}^{m_h}$ and equivariant latent variables $\boldsymbol{z}_i^{\boldsymbol{v}} \in \mathbb{R}^{m_v \times 3}$. Symbols $h, \boldsymbol{v}, z^h, \boldsymbol{z}^{\boldsymbol{v}}$ without subscripts refer to that variable for all nodes in a general sense. For column vectors $a \in \mathbb{R}^c$ and $b \in \mathbb{R}^c$, we use $a \odot b \in \mathbb{R}^c$ to denote pointwise multiplication and $\operatorname{diag}\{a\} \in \mathbb{R}^{c \times c}$ to denote a matrix whose diagonal is a and zero otherwise.

Equivariant Features For Coordinates Predictions. The equivariant nature makes it difficult to predict absolute coordinates directly. Many existing works (Gebauer et al., 2019; Anonymous, 2022; Xu et al., 2021) tackle this problem by encoding coordinate information as invariant node features and predicting invariant quantities such as distances and edge angles. However, these indirect methods are either computationally intensive (need transformation to local coordinate system (Anonymous, 2022)) or introduce extra error from the second nonconvex optimization (for translating distance matrices into absolute coordinates (Xu et al., 2021)). Instead, we propose to generate absolute coordinates directly while preserving equivariance. To generate equivariant coordinates, only leveraging invariant features is not enough: we cannot produce an equivariant quantity arbitrarily by combining invariant quantities. Therefore, in addition to invariant node features, we need to introduce extra equivariant node features that can be directly used for composing equivariant coordinates. We use notations h for invariant features and $\boldsymbol{v}$ for equivariant features.

Vector Neurons. Classical fully connected neural networks or MLPs cannot preserve equivariance and thus is not suitable for transforming equivariant features $\boldsymbol{v}$. In this regard, vector neuron networks or VN-MLP (Deng et al., 2021) propose a ReLU-like nonlinear function for equivariant features. Concretely, given an equivariant input $\boldsymbol{v} \in \mathbb{R}^{n_v \times 3}$, Vector-ReLU learns two weight matrices $W \in \mathbb{R}^{n'_v \times n_v}$ and $U \in \mathbb{R}^{n'_v \times n_v}$ to map $\boldsymbol{v}$ to output $\boldsymbol{v}' \in \mathbb{R}^{n'_v \times 3}$ via

$$\boldsymbol{q} = W \cdot \boldsymbol{v} \in \mathbb{R}^{n'_v \times 3}, \quad \boldsymbol{k} = U \cdot \boldsymbol{v} \in \mathbb{R}^{n'_v \times 3}, \tag{4a}$$

$$\boldsymbol{v}' = \boldsymbol{q} - \text{diag}\left\{ \mathbb{1}_{\langle \boldsymbol{q}, \boldsymbol{k} \rangle < 0} \odot \langle \boldsymbol{q}, \frac{\boldsymbol{k}}{\|\boldsymbol{k}\|} \rangle \right\} \cdot \frac{\boldsymbol{k}}{\|\boldsymbol{k}\|}, \tag{4b}$$

where $\langle \boldsymbol{q}, \boldsymbol{k} \rangle \in \mathbb{R}^{n'_v}$ is the inner product in the last axis, $\mathbb{1}$ is the indicator function, and $\|\boldsymbol{k}\| \in \mathbb{R}^{n'_v}$ is the norm of $\boldsymbol{k}$ over the last axis. It is easy to verify that $\boldsymbol{v}'$ is equivariant, since both $\boldsymbol{q}$ and $\boldsymbol{k}$ are linear combinations of equivariant input $\boldsymbol{v}$ while coefficients $\langle \boldsymbol{q}, \boldsymbol{k} \rangle$ is invariant. Intuitively, Vector-ReLU projects $\boldsymbol{q}$ to the orthogonal plane of a learnable direction $\boldsymbol{k}$ if $\boldsymbol{q}$ lies in the other side of the plane, which is analogous to the cutoff in classic ReLU. This nonlinearity enhances the expressive power while preserving the equivariance. We use VN-MLP to denote a neural network stacked by multiple Vector-ReLU units.

Mixed-Features Message Passing. Now we are ready to introduce our Mixed-Features Message Passing (MF-MP) scheme. MF-MP performs message passing for invariant features h and equivariant features $\boldsymbol{v}$ simultaneously, and in each step the two types of features are properly mixed so that 1) their respective invariance and equivariance properties are preserved, and 2) one type of feature helps the update of the other type and vice versa.

In the first step, invariant features $h \in \mathbb{R}^{n_h}$ and equivariant features $\boldsymbol{v} \in \mathbb{R}^{n_v \times 3}$ are transformed and mixed to construct new expressive intermediate features $h', h'', \boldsymbol{v}'$ by

$$h'_j = \phi_1(h_j, \|\text{VN-MLP}_1(\boldsymbol{v_j})\|) \in \mathbb{R}^{n_h}, \tag{5a}$$

$$h''_j = \phi_2(h_j, \|\text{VN-MLP}_2(\boldsymbol{v_j})\|) \in \mathbb{R}^{n_v}, \tag{5b}$$

$$\boldsymbol{v}'_j = \text{diag}\{\phi_3(h_j)\} \cdot \text{VN-MLP}_3(\boldsymbol{v}_j) \in \mathbb{R}^{n_v \times 3}. \tag{5c}$$

Next, point convolution (Thomas et al., 2018; Schütt et al., 2017; 2021) is applied to linearly transform the mixed features $h', h'', \boldsymbol{v}'$ into messages:

$$m^h_{i \leftarrow j} = \text{Ker}_1(\|\boldsymbol{r}_{i,j}\|) \odot h'_j, \tag{6a}$$

$$\begin{aligned} \boldsymbol{m}^{\boldsymbol{v}}_{i \leftarrow j} &= \text{diag}\left\{\text{Ker}_2(\|\boldsymbol{r}_{i,j}\|)\right\} \cdot \boldsymbol{v}'_j \\ &\quad + \left(\text{Ker}_3(\|\boldsymbol{r}_{i,j}\|) \odot h''_j\right) \cdot \boldsymbol{r}^\top_{i,j}, \end{aligned} \tag{6b}$$

where $\boldsymbol{r}_{i,j} = \boldsymbol{r}_i - \boldsymbol{r}_j$ is the relative displacement, Ker are learnable kernels such as RBFs that transform a scalar distance into a multi-dimensional output vector using different shape parameters, making the messages geometry-aware. Intuitively, it reflects discrete levels of physical interactions (short-range, long-range) at different distances. More details on Ker are given in Appendix B.

Finally, Gated Recurrent Units (GRU) (Li et al., 2015) and VN-MLP are applied as powerful nonlinear transformations to update the node features with the messages:

$$\tilde{h}_i = \text{GRU}(h_i, \sum_{j \in N(i)} m^h_{i \leftarrow j}), \tag{7a}$$

$$\tilde{\boldsymbol{v}}_i = \text{VN-MLP}_4(\boldsymbol{v}_i, \sum_{j \in N(i)} \boldsymbol{m}^{\boldsymbol{v}}_{i \leftarrow j}). \tag{7b}$$

Here $N(i)$ stands for neighbors of i. Our Mixed-Features Message Passing (MF-MP) above effectively mixes invariant and equivariant features in each step to help the update of each other with powerful nonlinear functions. Proof of MF-MP's equivariance w.r.t. E(3) is included in Appendix A. We also discuss how it relates to and differs from Tensor Field Networks (Thomas et al., 2018) in appendix B.

Now we describe details about the encoder and decoder of 3DLinker using MF-MP as building blocks. 3DLinker is a conditional latent generative model $p_{\theta,\psi}(G, \boldsymbol{R}|G_\text{F}, \boldsymbol{R}_\text{F})$ including an encoder $q_\psi(z^h, \boldsymbol{z^v}|G_\text{F}, G, \boldsymbol{R}_\text{F}, \boldsymbol{R})$, a decoder $p_\theta(G, \boldsymbol{R}|G_\text{F}, \boldsymbol{R}_\text{F}, z^h, \boldsymbol{z^v})$ and a prior $p(z^h, \boldsymbol{z^v}|G_\text{F}, \boldsymbol{R}_\text{F})$.

4.1. Encoder

The encoder $q_\psi(z^h, \boldsymbol{z^v}|G_\text{F}, G, \boldsymbol{R}_\text{F}, \boldsymbol{R})$ computes node-level latent distributions utilizing MF-MP. Initially, invariant features h are embeddings of node types and we let equivariant features $\boldsymbol{v} = 0$. After applying several times of MF-MP, we obtain the final node features $\tilde{h}$ and $\tilde{\boldsymbol{v}}$. The latent variables are sampled by $z^h_i \sim N(\mu^h_i, (\sigma^h_i)^2 I)$, $\boldsymbol{z}^{\boldsymbol{v}}_i \sim N(\boldsymbol{\mu}^{\boldsymbol{v}}_i, (\sigma^{\boldsymbol{v}}_i)^2 I)$ for linker nodes only, where the means and variances are computed from the final node features:

$$\forall i \in \mathcal{V}_\text{L},$$

$$\mu^h_i = \phi_4(\tilde{h}_i), \quad (\sigma^h_i)^2 = \phi_5(\tilde{h}_i), \tag{8a}$$

$$\boldsymbol{\mu}^{\boldsymbol{v}}_i = \text{VN-MLP}_5(\tilde{\boldsymbol{v}}_i), \quad (\sigma^{\boldsymbol{v}}_i)^2 = \phi_6(\tilde{h}_i). \tag{8b}$$

Note that for equivariant latent variables $\boldsymbol{z^v}$ the covariance $(\sigma^{\boldsymbol{v}})^2 I$ assign the same variance to x, y, z directions, which is the simplest way to preserve equivariance. Since fragments $(G_\text{F}, \boldsymbol{R}_\text{F})$ are given during generation, there is no need to sample their latent variables. Instead, we run the same (weight-sharing) MF-MP network again on fragments only, which gives another set of final node features $\hat{h}_i, \hat{\boldsymbol{v}}_i$ for fragment nodes. Latent variables of fragment nodes are deterministically obtained by

$$\forall i \in \mathcal{V}_\text{F},$$

$$z^h_i = \phi_7(\hat{h}_i), \quad \boldsymbol{z}^{\boldsymbol{v}}_i = \text{VN-MLP}_6(\hat{\boldsymbol{v}}_i). \tag{9a}$$

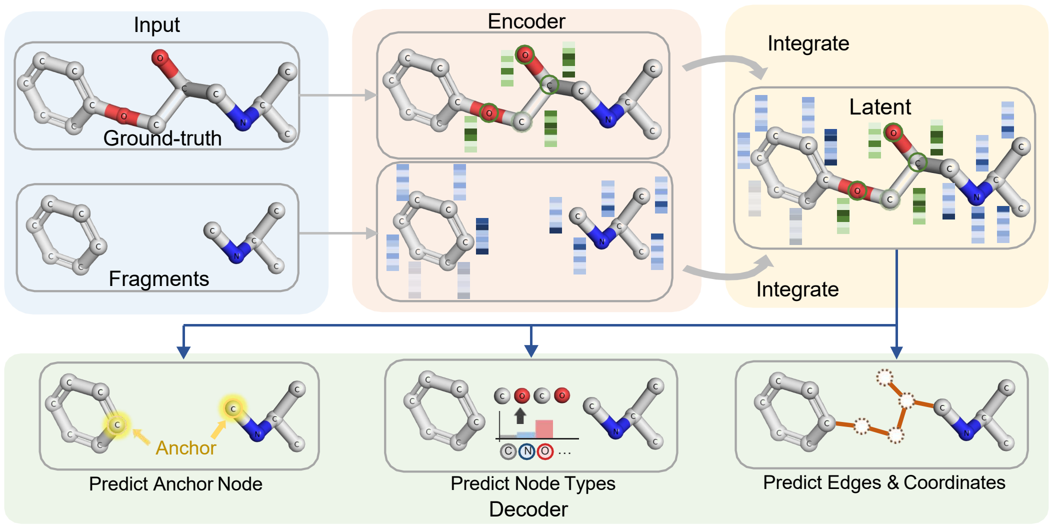

Figure 2: Illustration of overall encoding and decoding process. For encoding, ground truth is sent into a MF-MP encoder to get node-level representations. Those representations of nodes in fragments are discarded and replaced by representations that are computed separately on the fragments graph only. For decoding, two anchor nodes are predicted as the binding sites for the linker. Node Types of the linker are simultaneously predicted before linking. With two anchor nodes and node types of the linker, edges and coordinates are sequentially predicted, as demonstrated in Figure 3.

4.2. Decoder

The decoder $p_\theta(G, \boldsymbol{R}|G_{\text{F}}, \boldsymbol{R}_{\text{F}}, z^h, \boldsymbol{z^v})$ constructs $(G, \boldsymbol{R})$ from fragments $(G_{\text{F}}, \boldsymbol{R}_{\text{F}})$ in a sequential manner. In the decoding process we incorporate the valency rules of molecules by masking out impossible edges and anchor nodes. The decoding process consists of the following steps:

(1) **Anchor Node Prediction:** predict anchor nodes $a = (a_1, a_2)$ for the two fragments. These two anchor nodes are served as the binding points for linker to connect.

(2) **Node Type Prediction:** predict node type X for all linker nodes.

(3) **Edge and Coordinate Prediction:** Put the two anchor nodes in a queue. Then do the following until the queue is empty:

   (i) Pop a node f from the queue and define it as the current *focus node*.

   (ii) Predict an edge between the focus node f and another node. The connected node i is added to the queue. If node i is a linker node and is connected to the existing graph G_t for the first time, predict its coordinates $\boldsymbol{r}_i$.

   (iii) Repeat (ii) and (iii) until an artificial stop node is connected. Update the coordinates of all nodes in the current linker. The focus node f is then marked as closed, which cannot be added to the queue or connected anymore. Then go back to (i).

Mathematically we factorize the joint probability into:

$$p_\theta(G, \boldsymbol{R}|G_{\text{F}}, \boldsymbol{R}_{\text{F}}, z^h, \boldsymbol{z^v}) = p_\theta(\mathcal{E}, X, \boldsymbol{R}|\mathcal{E}_{\text{F}}, X_{\text{F}}, \boldsymbol{R}_{\text{F}}, z^h, \boldsymbol{z^v})$$
$$= \underbrace{p_\theta(a_1, a_2|z^h, \boldsymbol{z^v})}_{\text{Anchor}} \cdot \underbrace{p_\theta(X|z^h)}_{\text{Node Types}}$$
$$\cdot \underbrace{\prod_{t=0}^{T-1} p_\theta(\mathcal{E}_{t+1}, \boldsymbol{R}_{t+1}|\mathcal{E}_t, \boldsymbol{R}_t, X, a_1, a_2, z^h, \boldsymbol{z^v})}_{\text{Edges and Coordinates}}, \tag{10}$$

where $\mathcal{E}_T = \mathcal{E}$ and $\boldsymbol{R}_T = \boldsymbol{R}$. Details of each component are explained in the following.

Anchor Node Prediction. To jointly predict two anchor nodes, we further factorize the joint probability into p_{a_1}, the probability of anchor a_1 on the first fragment $G_{\text{F},1}$, and p_{a_2}, the probability of anchor a_2 on the second fragment $G_{\text{F},2}$ conditioning on a_1:

$$\begin{aligned} p_\theta(a_1, a_2|z^h, \boldsymbol{z^v}) = p_\theta(a_1|\{z_i^h, \boldsymbol{z}_i^{\boldsymbol{v}}|i \in \mathcal{V}_{\text{F},1}) \\ \cdot\, p_\theta(a_2|z_{a_1}^h, \boldsymbol{z}_{a_1}^{\boldsymbol{v}}, \{z_i^h, \boldsymbol{z}_i^{\boldsymbol{v}}|i \in \mathcal{V}_{\text{F},2}). \end{aligned} \tag{11}$$

Concretely, each node on the first fragment will get a score $c_i = \phi_8(z_i^h, \|A_1 \cdot \boldsymbol{z}_i^{\boldsymbol{v}}\|)$, where $A_1 \in \mathbb{R}^{n_v \times n_v}$ is a learnable linear transformation. The scores c_i are then passed to a softmax to compute the anchor probability for nodes of the first fragment: $p_{a_1} = \exp(c_{a_1})/(\sum_{i\in\mathcal{V}_{\text{F},1}} \exp(c_i))$. Then the latent variables of this predicted anchor node as well as nodes of the second fragment are used to compute another group of scores $c_i' = \phi_9(z_i^h, \|A_1 \cdot \boldsymbol{z}_i^{\boldsymbol{v}}\|, z_{a_1}^h, \|A_1 \cdot \boldsymbol{z}_{a_1}^{\boldsymbol{v}}\|)$, and the probability of the second anchor is $p_{a_2} = \exp(c'_{a_2})/(\sum_{i \in \mathcal{V}_{\mathrm{F},2}} \exp(c'_i))$.

Node Type Prediction. Node types of the linker are directly predicted using their latent variables. We leverage the self-attention mechanism (Vaswani et al., 2017) to obtain new node features, which are then passed to an MLP to get the logits of node types. After node types are sampled, the types' embeddings are concatenated to the corresponding latent variables z^h for the latter procedures.

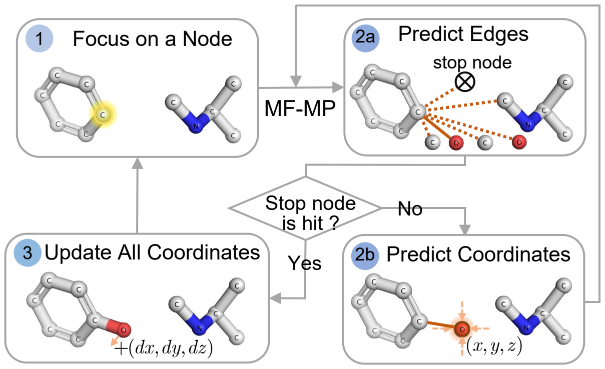

Figure 3: Illustration of sequential predictions of edges and coordinates. We first pick up on a node to focus. Then we sample an edge between the focus node and other nodes (including an artificial stop node). If a linker node is first connected to the existing graph, its coordinates will be predicted. Each time before prediction MF-MP is applied to capture information from the existing graph. We keep adding edges until the stop node is selected, and then coordinates of all link nodes in the existing graph will be simultaneously updated. We then refocus on a new node and repeat. The procedure continues until all nodes in the linker have been focused.

Edge and Coordinate Prediction. The edge and coordinate generation path $(\mathcal{E}_0, \boldsymbol{R}_0) \to (\mathcal{E}_1, \boldsymbol{R}_1) ... \to (\mathcal{E}, \boldsymbol{R})$ is defined by a sequence of node focusing, edge prediction and coordinate prediction procedures. The node focusing and edge connecting order is pre-determined by a Breath-First Search to enable teacher-forcing training. When a focus node f is picked, we add new edges to it until connecting to the stop node. Concretely, at each step we first apply MF-MP (with different weights from that in the encoder) to obtain updated node features $\tilde{z}^h, \tilde{\boldsymbol{z}}^{\boldsymbol{v}}$ from the initial latent variables $z^h, \boldsymbol{z}^{\boldsymbol{v}}$. Then we compute the probability for edge $\mathcal{E}_{i,f}$ by

$$s_{i,f} = \phi_{10}(\tilde{z}_i^h, \tilde{z}_f^h, \|A_2 \cdot \tilde{\boldsymbol{z}}_i^{\boldsymbol{v}}\|, \|A_2 \cdot \tilde{\boldsymbol{z}}_f^{\boldsymbol{v}}\|, \sum_{j \in \mathcal{V}} \tilde{z}_j^h, \sum_{j \in \mathcal{V}} z_j^h).$$

$$p(\mathcal{E}_{i,f}) = \frac{\exp(s_{i,f})}{\sum_{j \in \mathcal{V}} \exp(s_{j,f})}. \tag{12}$$

To predict coordinates, we define $\boldsymbol{\Omega}_i(\tilde{z}^h, \tilde{\boldsymbol{z}}^{\boldsymbol{v}}, \boldsymbol{R}_t, \boldsymbol{r})$ that outputs equivariant coordinates $\tilde{\boldsymbol{r}}_i$ for node i:

$$p_{i,j} = \phi_{11}(\tilde{z}_i^h, \tilde{z}_j^h, \langle A_3 \cdot \tilde{\boldsymbol{z}}_i^{\boldsymbol{v}}, A_4 \cdot \tilde{\boldsymbol{z}}_j^{\boldsymbol{v}} \rangle), \tag{13a}$$

$$q_{i,j} = \phi_{12}(\tilde{z}_i^h, \tilde{z}_j^h, \langle A_5 \cdot \tilde{\boldsymbol{z}}_i^{\boldsymbol{v}}, A_6 \cdot \tilde{\boldsymbol{z}}_j^{\boldsymbol{v}} \rangle), \tag{13b}$$

$$\tilde{\boldsymbol{r}}_i = \boldsymbol{r} + \sum_{j \in \mathcal{V}_t} p_{i,j}(\boldsymbol{r}_j - \boldsymbol{r}) + \text{VN-MLP}_7\left(\sum_{j \in \mathcal{V}_t} q_{i,j} \text{VN-MLP}_8(\tilde{\boldsymbol{z}}_i^{\boldsymbol{v}}, \tilde{\boldsymbol{z}}_j^{\boldsymbol{v}}) \right) \tag{13c}$$

for any generic coordinates $\boldsymbol{r}$ called reference point. Here VN-MLP taking two inputs means concatenation along the first axis (n_v). The idea is to compute pair-wise interactions (13a, 13b) and predict a deviation from reference point $\boldsymbol{r}$ (13c). If a linker node i is first connected to the graph, we use the mass center of the current graph $\bar{\boldsymbol{r}}_t = \sum_{j \in \mathcal{V}_t} \boldsymbol{r}_j / |\mathcal{V}_t|$ as the reference point and predict its absolute coordinates by $\boldsymbol{r}_i = \boldsymbol{\Omega}_i^{\text{pred}}(\tilde{z}^h, \tilde{\boldsymbol{z}}^{\boldsymbol{v}}, \boldsymbol{R}_t, \bar{\boldsymbol{r}}_t)$; once the stop node is chosen, all linker nodes i in the current graph will update their coordinates using their current coordinates as reference points, i.e., $\boldsymbol{r}'_i = \boldsymbol{\Omega}_i^{\text{updt}}(\tilde{z}^h, \tilde{\boldsymbol{z}}^{\boldsymbol{v}}, \boldsymbol{R}_t, \boldsymbol{r}_i)$. Note that $\boldsymbol{\Omega}^{\text{pred}}$ and $\boldsymbol{\Omega}^{\text{updt}}$ have distinct network weights.

4.3. Training Using ELBO

Variational autoencoder (Kingma & Welling, 2013) is trained by maximizing the Evidence Lower Bound (ELBO):

$$\begin{aligned} L(\theta, \phi, \psi) = \mathbb{E}_{z \sim q_\psi} \log p_\theta(G, \boldsymbol{R} | G_{\mathrm{F}}, \boldsymbol{R}_{\mathrm{F}}, z^h, \boldsymbol{z}^{\boldsymbol{v}}) \\ - \beta D_{\mathrm{KL}}(q_\psi \| p), \end{aligned} \tag{14}$$

where the prior $p(z^h, \boldsymbol{z}^{\boldsymbol{v}} | G_{\mathrm{F}}, \boldsymbol{R}_{\mathrm{F}})$ simply takes standard Gaussian for all linker nodes. The reconstruction error term $\mathbb{E}_{z \sim q_\psi} \log p_\theta(G, \boldsymbol{R} | G_{\mathrm{F}}, \boldsymbol{R}_{\mathrm{F}}, z^h, \boldsymbol{z}^{\boldsymbol{v}})$ is approximated by one Monte-Carlo sampling, and we apply teacher forcing (Kolen & Kremer, 2001) for anchor, node type and edge prediction following the pre-determined order. The loss of anchor node prediction, node type prediction and edge prediction are standard cross entropy while for loss of coordinate prediction we use log-MSE used by (Yu, 2020).

4.4. Generation

During generation, a maximum number of linker nodes is set and we sample this maximum number of latent variables $z^h, \boldsymbol{z}^{\boldsymbol{v}}$ for linker nodes (though some nodes might never be included). Then the generation follows the same procedure as the decoder except that there is no teacher forcing.

5. Experiments

5.1. Experiment Setup

Dataset. To evaluate our model, we choose a subset of ZINC (Sterling & Irwin, 2015). For each molecule, we

Table 1: Performance metrics for generated molecules on ZINC dataset. Uncertainty is estimated by 3-σ deviation.

Metrics (%)	Valid	Recover	Pass 2D filters	S_{tani}	RMSD (Å)	Unique	Novel
3DLinker (given anchor)	**99.20**$_{\pm 0.04}$	**95.24**$_{\pm 1.17}$	90.39$_{\pm 0.11}$	**55.06**$_{\pm 0.09}$	**0.081**$_{\pm 0.005}$	29.20$_{\pm 0.11}$	32.16$_{\pm 0.12}$
3DLinker	**98.68**$_{\pm 0.03}$	**93.75**$_{\pm 0.36}$	90.32$_{\pm 0.18}$	**54.83**$_{\pm 0.08}$	**0.081**$_{\pm 0.004}$	29.34$_{\pm 0.17}$	32.53$_{\pm 0.12}$
DeLinker+ConfVAE	98.35$_{\pm 0.07}$	80.35$_{\pm 2.58}$	89.92$_{\pm 0.53}$	52.86$_{\pm 0.04}$	1.345$_{\pm 0.028}$	44.53$_{\pm 0.42}$	39.53$_{\pm 0.58}$
GraphAF+ConfVAE	34.25$_{\pm 0.17}$	21.23$_{\pm 1.82}$	82.00$_{\pm 2.21}$	38.00$_{\pm 0.14}$	1.263$_{\pm 0.124}$	84.06$_{\pm 0.31}$	78.33$_{\pm 0.12}$
GraphVAE+ConfVAE	63.07$_{\pm 0.59}$	1.40$_{\pm 1.37}$	86.17$_{\pm 0.26}$	51.31$_{\pm 0.07}$	1.523$_{\pm 0.478}$	43.59$_{\pm 0.35}$	91.59$_{\pm 1.47}$
SyntaLinker+ConfVAE	80.14$_{\pm 0.31}$	85.47$_{\pm 2.46}$	97.41$_{\pm 0.11}$	54.80$_{\pm 0.05}$	1.402$_{\pm 0.016}$	41.51$_{\pm 0.75}$	13.17$_{\pm 0.27}$
Gen3D	75.92$_{\pm 0.10}$	37.99$_{\pm 0.67}$	81.41$_{\pm 0.12}$	48.67$_{\pm 0.06}$	1.415$_{\pm 0.272}$	49.86$_{\pm 0.49}$	64.17$_{\pm 0.21}$

perform 20 times of MMFF force field optimization using RDKit (Landrum) and choose the one with the lowest energy as the ground truth. Following the same procedure from (Hussain & Rea, 2010), the (fragments, linker) pairs are produced by enumerating all double cuts of acyclic single bonds that are not within any functional groups. In total, we obtain 365,749 (fragments, linker, coordinates) triplets and randomly split them into training (365,039), validation (351) and test (358).

Evaluation. We evaluate the generated molecules for multiple 2D (graph) and 3D (coordinates) metrics, including the standard ones such as validity, uniqueness and novelty (Brown et al., 2019). In addition, we also evaluate the percentage of generated molecules passing 2D property filters, including synthetic accessibility (Ertl & Schuffenhauer, 2009), ring aromaticity, and pan-assay interference compounds (PAINS) (Baell & Holloway, 2010). After filtering by validity and 2D property filters, the recovery rate is calculated to report the percentage of generated molecules that perfectly recover the ground truth molecule graphs. We also compare Tanimoto similarity using the Morgan fingerprint provided by RDKit, which estimates the similarity between ground-truth and generated molecular graphs. To evaluate the quality of the 3D structures, the predicted 3D structures are compared to the ground truth using root-mean-square deviation (RMSD). Note that RMSD is only computed for generated molecules that perfectly recover the ground truth molecular graphs (including their isomorphic variants), since only recovered molecules have atom-to-atom alignment to ground truth. Also note that We do not apply MMFF to the generated structures since hydrogen is not explicitly included in the dataset for computational efficiency. Following DeLinker, we compute another 3D metric, named shape-and-color similarity score (SC_{RDKit}). Appendix C contains more details about the evaluation standards.

Baselines. Though there are works of molecular graph generative models and molecular geometry prediction given molecular graph, they rarely focus on fragment linking or jointly modeling both graph and geometry. Existing molecule generative models either do not work on conditional (linker) generation, or cannot predict 3D coordinates. Therefore, we implement multiple baselines by adapting multiple generative models to conditional generative models to generate 2D linker graphs given two molecular fragments. The generated graphs are then taken by a molecular geometry prediction model, ConfVAE (Xu et al., 2021), to predict the 3D coordinates of each atom. Our baselines include DeLinker+ConfVAE, GraphAF+ConfVAE ,GraphVAE+ConfVAE, SyntaLinker+ConfVAE and Gen3D. DeLinker and SyntaLinker are existing baselines for conditional linker generation. GraphAF is an autoregressive flow model, and GraphVAE is a VAE-based model with graph-level encodings. Gen3D is an autoregressive model using EGNN that can jointly generate graphs and coordinates. The latter three are adapted to conditional graph generation. For ConfVAE, we modify its decoder flow model to predict linker coordinates conditionally. Please see appendix C for implementation details.

Table 2: SC_{RDKit} score distribution (%) and averaged score on ZINC.

Metrics	SC_{RDKit} Fragments			
	> 0.7	> 0.8	> 0.9	Average
3DLinker (given anchor)	**43.10**	**16.09**	**2.60**	**0.684**
3DLinker	42.55	15.85	2.49	0.683
DeLinker+ConfVAE	39.96	13.39	1.93	0.675
GraphAF+ConfVAE	19.33	3.36	0.32	0.624
GraphVAE+ConfVAE	13.17	2.15	0.00	0.601

5.2. Results

We trained 3DLinker for 20 epochs using Adam optimizer with a learning rate 0.006, batch size 48 and KL trade-off $\beta = 0.6$. Training details for other baselines are included in Appendix C. Each model generates 250 samples per two fragments, which leads to in total $250 \times 358 = 89500$ samples. We conduct such generations three times independently for uncertainty estimation of metrics in the main table 1. Note that since DeLinker takes anchor nodes as known ground truth, we add another comparison where anchor nodes are known to 3DLinker when generation, denoted as 3DLinker (given anchor). Results in Table 1 and Table 2 show that 3DLinker could generate valid and similar linker graph structures with a higher recovery rate, and at the same time achieve accurate predictions of the 3D coordinates of each atom. An interesting observation is that although focusing on 3D structures, 3DLinker achieves superior recovering

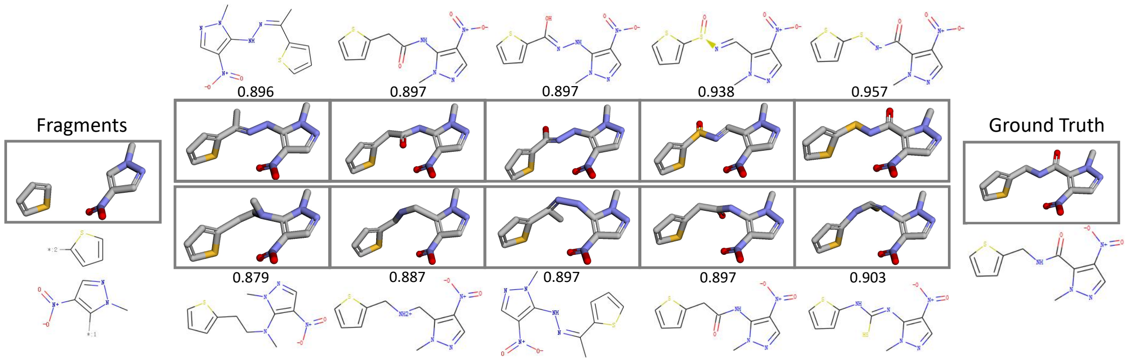

Figure 4: An example of fragment linking. The top-5 similar by SC_{RDKit} Fragments proposed by 3DLinker (first row) and DeLinker+ConfVAE (second row) are shown. Generations from 3DLinker are more realistic and similar to ground truth in terms of SC_{RDKit} and 3D geometry.

accuracy of the 2D molecular graphs. Our interpretation is that incorporating 3D constraints benefits the reconstruction of 2D graphs, though it might influence the diversity (low novelty and uniqueness). This also explains why the novelty and uniqueness are relatively low because the 3D constraints significantly reduce the valid chemical compound structures. In addition, GraphAF and GraphVAE are not able to obey valency rules during training, which explains their low valid and recovery rates. In terms of 3D structure predictions, the low RMSD of 3DLinker demonstrates the effectiveness of both equivariant features and coordinate update strategies. We also notice that since SyntaLinker has no constraints on chemical valency or whether output contains the original fragments, it turns out to have low validity. Nevertheless, it has a higher rate of passing ring aromaticity filter (one of 2D filters), which might be because operating on SMILES does not easily break aromaticity, while operating on nodes and edges sometimes does. GEN3D turns out to have a poor RMSD, and we find it is partially because GEN3D predicts distance matrices between atoms: a small perturbation to the distance matrix may lead to significant changes of conformation in long-chain structures, and the error scales up quickly with the increasing of number of nodes. This may also explain the poor performance of ConfVAE, which also makes predictions based on distance matrices.

5.3. Ablation Study

In the ablation study, we focus on two key components of 3DLinker: (1) equivariant features vs. invariant features alone; (2) update of all coordinates until the generation finishes, instead of fixing them in the 3D space one by one. Results display a significant performance decrease after removing either equivariant features or coordinate updates, especially for the RMSD and recovery rate. See Appendix D for details.

Table 3: Root Mean Squared Error of QED prediction on ZINC dataset.

Metrics	$RMSE_{QED}$
3DLinker	0.0833
DeLinker+ConfVAE	0.1077
GraphVAE+ConfVAE	0.1179

5.4. Molecular Property Prediction

We show a downstream task of molecule property prediction. We use the learned latent variables from VAE models to predict the Quantitative Estimate of Drug-Likeness (QED) by a gated sum:

$$\text{QED} = \sigma\left(\sum_{i\in G} \sigma(\phi_{13}(z_i^h, \|V\boldsymbol{z}_i^{\boldsymbol{v}}\|))\phi_{14}(z_i^h, \|U\boldsymbol{z}_i^{\boldsymbol{v}}\|)\right), \tag{15}$$

where ϕ_{13} and ϕ_{14} are two separate neural networks, V, U are two linear transform matrices and $\sigma(\cdot)$ is a sigmoid function. QED is widely adopted to quantify the potential for a small molecular to be a drug while the function of molecular linkers is to connect two existing drugs. Here we adopt QED to check whether 3DLinker produces biological and biochemical meaningful molecules. Training and test sets are in the same setup as in Section 5.1, with QED scores computed by RDKit. For each pre-trained model, we train a QED predictor for 20 epochs with a learning rate of 0.006 and batch size of 48.

As shown in Table 3, 3DLinker achieves the lowest Root Mean Square Error (RMSE) among all the models, suggesting a good expressing power of its learned latent representations.

5.5. Visualization

In the end, we present multiple examples of linker design by visualizations (see more examples of visualization in Appendix E.). In Figure 4, we show the top-5 molecules with highest SC_{RDKit} generated by 3DLinker (first row in the middle) and DeLinker+ConfVAE (second row in the middle). It is obvious that molecules from 3DLinker are generally more similar to the ground truth 2D chemical graph, and have a better spatial alignment with the ground truth 3D structure. In practice, when the ground truth is unknown, the "best" linker could be estimated by the generation likelihood.

6. Conclusions

We developed 3DLinker, a conditional variational autoencoder that is able to jointly model graph and 3D representations, predict anchor nodes and samples linkers. Experiments show that 3DLinker is able to generate linkers with both a high recovery rate and precise geometry.

There are still limitations to be considered in the future. First, models should be able to sample number of linker nodes instead of setting a maximal number of nodes in advance. Second, though it is known that the spatial configuration of fragments should not be disturbed, in practice slight differences exist between different linkers, which is an avenue for future works to take into account.

Software and Data

We implement 3DLinker based on the released code of DeLinker (https://github.com/fimrie/DeLinker). Our code and data are available at https://github.com/GraphPKU/3DLinker.

References

Anonymous. An autoregressive flow model for 3d molecular geometry generation from scratch. In *Submitted to The Tenth International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=C03Ajc-NS5W. under review.

Baell, J. B. and Holloway, G. A. New substructure filters for removal of pan assay interference compounds (pains) from screening libraries and for their exclusion in bioassays. *Journal of medicinal chemistry*, 53(7):2719–2740, 2010.

Brown, N., Fiscato, M., Segler, M. H., and Vaucher, A. C. Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108, 2019.

Deng, C., Litany, O., Duan, Y., Poulenard, A., Tagliasacchi, A., and Guibas, L. Vector neurons: A general framework for so (3)-equivariant networks. *arXiv preprint arXiv:2104.12229*, 2021.

Ertl, P. and Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1(1):1–11, 2009.

Fuchs, F. B., Worrall, D. E., Fischer, V., and Welling, M. Se (3)-transformers: 3d roto-translation equivariant attention networks. *arXiv preprint arXiv:2006.10503*, 2020.

Gebauer, N. W., Gastegger, M., and Schütt, K. T. Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. *arXiv preprint arXiv:1906.00957*, 2019.

Griffiths, D. J. and Schroeter, D. F. *Introduction to quantum mechanics*. Cambridge University Press, 2018.

Hussain, J. and Rea, C. Computationally efficient algorithm to identify matched molecular pairs (mmps) in large data sets. *Journal of chemical information and modeling*, 50 (3):339–348, 2010.

Ichihara, O., Barker, J., Law, R. J., and Whittaker, M. Compound design by fragment-linking. *Molecular Informatics*, 30(4):298–306, 2011.

Imrie, F., Bradley, A. R., van der Schaar, M., and Deane, C. M. Deep generative models for 3d linker design. *Journal of chemical information and modeling*, 60(4):1983–1995, 2020.

Imrie, F., Hadfield, T. E., Bradley, A. R., and Deane, C. M. Deep generative design with 3d pharmacophoric constraints. *bioRxiv*, 2021.

Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

Jin, W., Barzilay, R., and Jaakkola, T. Hierarchical generation of molecular graphs using structural motifs. In *International Conference on Machine Learning*, pp. 4839–4848. PMLR, 2020.

Jing, B., Eismann, S., Suriana, P., Townshend, R. J., and Dror, R. Learning from protein structure with geometric vector perceptrons. *arXiv preprint arXiv:2009.01411*, 2020.

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

Klicpera, J., Groß, J., and Günnemann, S. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020.

Klon, A. E. *Fragment-Based Methods in Drug Discovery*. Springer, 2015.

Kolen, J. F. and Kremer, S. C. *A field guide to dynamical recurrent networks*. John Wiley & Sons, 2001.

Landrum, G. Rdkit: Open-source cheminformatics. http://www.rdkit.org/. Accessed: 2022/1/13.

Landrum, G. A., Penzotti, J. E., and Putta, S. Feature-map vectors: a new class of informative descriptors for computational drug discovery. *Journal of computer-aided molecular design*, 20(12):751–762, 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.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

Li, Y., Vinyals, O., Dyer, C., Pascanu, R., and Battaglia, P. Learning deep generative models of graphs. *arXiv preprint arXiv:1803.03324*, 2018.

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.

Liu, Y., Wang, L., Liu, M., Zhang, X., Oztekin, B., and Ji, S. Spherical message passing for 3d graph networks. *arXiv preprint arXiv:2102.05013*, 2021.

Luo, S., Guan, J., Ma, J., and Peng, J. A 3d generative model for structure-based drug design. *Advances in Neural Information Processing Systems*, 34, 2021.

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

Polishchuk, P. G., Madzhidov, T. I., and Varnek, A. Estimation of the size of drug-like chemical space based on gdb-17 data. *Journal of computer-aided molecular design*, 27(8):675–679, 2013.

Putta, S., Landrum, G. A., and Penzotti, J. E. Conformation mining: an algorithm for finding biologically relevant conformations. *Journal of medicinal chemistry*, 48(9): 3313–3318, 2005.

Roney, J. P., Maragakis, P., Skopp, P., and Shaw, D. E. Generating realistic 3d molecules with an equivariant conditional likelihood model. 2021.

Satorras, V. G., Hoogeboom, E., Fuchs, F. B., Posner, I., and Welling, M. E (n) equivariant normalizing flows for molecule generation in 3d. *arXiv preprint arXiv:2105.09016*, 2021a.

Satorras, V. G., Hoogeboom, E., and Welling, M. E (n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021b.

Schütt, K., Kindermans, P.-J., Felix, H. E. S., Chmiela, S., Tkatchenko, A., and Müller, K.-R. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *NIPS*, 2017.

Schütt, K. T., Unke, O. T., and Gastegger, M. Equivariant message passing for the prediction of tensorial properties and molecular spectra. *arXiv preprint arXiv:2102.03150*, 2021.

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.

Simm, G., Pinsler, R., and Hernández-Lobato, J. M. Reinforcement learning for molecular design guided by quantum mechanics. In *International Conference on Machine Learning*, pp. 8959–8969. PMLR, 2020a.

Simm, G. N., Pinsler, R., Csányi, G., and Hernández-Lobato, J. M. Symmetry-aware actor-critic for 3d molecular design. *arXiv preprint arXiv:2011.12747*, 2020b.

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.

Sterling, T. and Irwin, J. J. Zinc 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11):2324–2337, 2015.

Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., and Riley, P. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Xu, M., Wang, W., Luo, S., Shi, C., Bengio, Y., Gomez-Bombarelli, R., and Tang, J. An end-to-end framework for molecular conformation generation via bilevel programming. *arXiv preprint arXiv:2105.07246*, 2021.

Yang, Y., Zheng, S., Su, S., Zhao, C., Xu, J., and Chen, H. Syntalinker: automatic fragment linking with deep conditional transformer neural networks. *Chemical science*, 11 (31):8312–8322, 2020.

You, J., Ying, R., Ren, X., Hamilton, W., and Leskovec, J. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International conference on machine learning*, pp. 5708–5717. PMLR, 2018.

Yu, R. A tutorial on vaes: From bayes' rule to lossless compression. *arXiv preprint arXiv:2006.10273*, 2020.

A. Proof of Invariance/Equivariance

In this section we prove that 3DLinker satisfies E(3) equivariance, namely for all $g \in$ E(3), we have $p_\theta(G, \pi(g)\boldsymbol{R}|G_{\text{F}}, \pi(g)\boldsymbol{R}_{\text{F}}) = p_\theta(G, \boldsymbol{R}|G_{\text{F}}, \boldsymbol{R}_{\text{F}})$. For simplicity, we denote E(3) invariance and equivariance as E(3)-inv and E(3)-eqv, and O(3) invariance and equivariance as O(3)-inv and O(3)-eqv, and translational invariance as T-inv.

We first prove $\tilde{h}$ is E(3)-inv while $\tilde{\boldsymbol{v}}$ is O(3)-eqv/T-inv after MF-MP (5, 6, 7)

Lemma A.1. *In MF-MP, $\tilde{h}$ is E(3)-inv and $\tilde{\boldsymbol{v}}$ is O(3)-eqv, if h is E(3)-inv and $\boldsymbol{v}$ is O(3)-eqv.*

Proof. First let us show that in (5), h', h'' are E(3)-inv and $\boldsymbol{v}'$ is O(3)-eqv/T-inv. Note that VN-MLP$(\boldsymbol{v})$ is E(3)-eqv and T-inv due to the properties of vector neurons, and thus the norm $\|$VN-MLP$(\boldsymbol{v})\|$ is O(3)-inv and T-inv (because O(3) preserves inner product), namely E(3)-inv. Therefore in (5a, 5b) h' and h'' are both E(3)-inv. Equation (5c) is essentially an O(3)-eqv VN-MLP$_{hv}(\boldsymbol{v}_j)$ scaled by E(3)-inv features diag$\{\phi_{hv}(h_j)\}$, and thus its output $\boldsymbol{v}'_j$ is also O(3)-eqv/T-inv.

Then in message function (6a, 6b), message m^h is E(3)-inv since kernel Ker is only a function of E(3)-inv distance $\|\boldsymbol{r}_{i,j}\|$. For (6b), The first term of RHS is a O(3)-eqv/T-inv features $\boldsymbol{v}'$ scaled by E(3)-inv kernel, which makes it O(3)-eqv/T-inv. Similarly for the second term is a O(3)-eqv/T-inv relative displacement $\boldsymbol{r}_{i,j}$ scaled by both E(3)-inv kernels and features. Therefore m^h is E(3)-inv and $\boldsymbol{m^v}$ is O(3)-eqv/T-inv.

Finally, in (7a) $\tilde{h}$ is E(3)-inv since all its inputs are E(3)-inv. In (7b) $\tilde{\boldsymbol{v}}$ is O(3)-eqv/T-inv due to vector neurons. □

Theorem A.2. *The generative model $p_\theta(G, \boldsymbol{R}|G_F, \boldsymbol{R}_F)$ (8, 9, 10) satisfies equivariance condition (3).*

Proof. By lemma A.1, the encoded latent variables z^h and $\boldsymbol{z^v}$ from (8, 9) is E(3)-inv and O(3)-eqv/T-inv respectively. In the decoding process, anchor nodes, node types and edges are all predicted from z^h or norm of $\boldsymbol{z^v}$. Thus the probability of graph G is E(3)-inv. The coordinates are predicted through (13). Note that $p_{i,j}$, $q_{i,j}$ in (13a, 13b) are E(3)-inv and $\boldsymbol{r}$ in (13c) is E(3)-eqv (mass center), which implies terms $\sum_{j\in\mathcal{V}_t} p_{i,j}(\boldsymbol{r}_j - \boldsymbol{r})$ and VN-MLP$_7\left(\sum_{j\in\mathcal{V}_t} q_{i,j}\text{VN-MLP}_8(\tilde{\boldsymbol{z}}^{\boldsymbol{v}}_i, \tilde{\boldsymbol{z}}^{\boldsymbol{v}}_j)\right)$ are all O(3)-eqv/T-inv. Finally we can conclude that $\boldsymbol{\Omega}_i$ is E(3)-eqv because an O(3)-eqv/T-inv quantity pluses an E(3)-eqv quantity $\boldsymbol{r}$ results in an E(3)-eqv quantity). Therefore $\boldsymbol{R}$ is E(3)-eqv. □

B. Point Convolution

In the message function (6), kernel Ker$(\|\boldsymbol{r}_{i,j}\|)$ assigns weights relying on distance $\|\boldsymbol{r}_{i,j}\|$. Concretely in our implementation, our kernel first applies Gaussian functions with 10 different means and perform a learnable affine transform:

$$\text{Gaussian} : \|\boldsymbol{r}_{i,j}\| \mapsto \begin{pmatrix} \exp\{-k(\|\boldsymbol{r}_{i,j}\| - \mu_1)\} \\ \exp\{-k(\|\boldsymbol{r}_{i,j}\| - \mu_2)\} \\ \cdots \\ \exp\{-k(\|\boldsymbol{r}_{i,j}\| - \mu_{10})\} \end{pmatrix}, \quad \text{Ker}(\|\boldsymbol{r}_{i,j}\|) = \text{Linear}(\text{Gaussian}(\|\boldsymbol{r}_{i,j}\|)) + \text{Bias}. \tag{16}$$

where $\mu_1 < \mu_2 < ... < \mu_{10}$ are hyper-parameters and k is a learnable parameter. The intuition behind is to capture the intensity of interactions between nodes (atoms) with different distances. The largest mean μ_{10} is basically the maximal correlation length: if two nodes are separated by distance beyond μ_{10}, their interaction is nearly neglectable.

Mathematically, message functions (6) can be seen as tensor products using spherical harmonics, as described in Tensor Field Networks (Thomas et al., 2018). In group representation theory, invariant features h and equivariant features $\boldsymbol{v}$ are called type-0 and type-1 tensors respectively, and the theory (Griffiths & Schroeter, 2018) tells us the correct way to construct new type-0 features $\tilde{h}$ or type-1 features $\tilde{\boldsymbol{v}}$ using type-l spherical harmonics $Y^l(\boldsymbol{r})$ and $h, \boldsymbol{v}$. In our case since we only have type-0 and type-1 features, $Y^0(\boldsymbol{r}) \propto 1$ and $Y^1 \propto \boldsymbol{r}$ are all we need, which explains the design of (6).

Note that there are several differences between our method and tensor products in Tensor Field Networks (TFN). First TFN is based on SO(3) equivariant, while we seek for O(3) equivariant. So terms like cross product are discarded since they violate mirror symmetry. Besides, we mix different types of features before convolution, in contrast to convolution on raw features. Finally, TFN uses simple activation functions like scaling with norm, while we leverage Vector Neuron for novel non-linearity.

C. Experiment Details

Some evaluation metrics. Validity is defined by percentage of generated molecules that both obey chemical constraints (valency) and successfully links two fragments into connected graphs. Invalid molecules are discarded for the following evaluations. Uniqueness means percentage of non-duplicate generated molecules: $\frac{\text{len(Set(generated molecules))}}{\text{len(generated molecules)}}$. Novelty refers to percentage of generated molecules whose linkers are not present in training set.

SC_{RDKit} uses two RDKit built-in functions as described in (Putta et al., 2005) and (Landrum et al., 2006) to compute color similarity scores between two 3D molecules based on the overlap of their pharmacophoric features. And the shape similarity score is a simple volumetric comparison between the two 3D molecules. Both scores are between 0 (no match) and 1 (perfect match), which are averaged to produce a final score between 0 and 1. Scores above 0.7 indicate a good match, while scores above 0.9 suggest an almost perfect match. Following DeLinker, SC_{RDKit} is measured only on fragments (we re-generate their coordinates together with the linker), which embodies the capability to generate linkers without disturbing fragments.

3DLinker. We train 3DLinker for 20 epochs with kl trade-off beta 0.6. Note that the anchor node prediction 11 is asymmetric to the permutation of a_1 and a_2, and thus we apply a permutation of two fragments to enhance our model.

GraphAF. Originally GraphAF is an autogressive flow model $p(G) = \prod_t p(G_{t+1}|G_t)$. To model a conditional probability $p(G|G_{\text{F}})$, all we need is to mask out loss of G_{F} and only compute loss starting from G_{F} to G. We trained GraphAF for 170 epochs, and other hyper-parameters are consistent with its source code (https://github.com/DeepGraphLearning/GraphAF).

GraphVAE. GraphVAE represents a graph G as node types F, adjacency A and edge types $\mathcal{E}$ and maps them into a graph-level representation z. Then z is decoded into $\tilde{F}, \tilde{A}, \tilde{\mathcal{E}}$ with an additional graph matching to compute loss. To modify it into a conditional generative models, a naive approach is to build a generative model for linker G_{L} only, and then predict the anchor nodes that connects to fragments. Concretely, let a_1, a_2 be anchor nodes of fragments, and b_1, b_2 be the corresponding anchor nodes of linker, the decoder model $p(G|G_{\text{F}}, z)$ is:

$$p(G|G_{\text{F}}, z) = p(G_{\text{L}}|z_{\text{F}}, z)p(a_1, a_2, b_1, b_2|z_{\text{F}}, z)p(\mathcal{E}_{a_1,b_1}, \mathcal{E}_{a_2,b_2}|z_{\text{F}}, z), \tag{17}$$

where z_F is a graph-level encoding of fragments and $\mathcal{E}_{a_1,b_1}, \mathcal{E}_{a_2,b_2}$ are two edges that connects fragments and linker. The realization is based on code provided by https://github.com/snap-stanford/GraphRNN. In experiments, all hyper-parameters are unchanged except KL trade-off beta is 0.6.

ConfVAE. ConfVAE is a VAE for geometry generation given graph $p(\boldsymbol{R}|G)$. It encodes graph G and distance matrix $D = \{d_{i,j} = \|\boldsymbol{r}_i - \boldsymbol{r}_j\| | i, j \in G\}$ into latent variables z, and uses a flow model f to decode distance matrix $D = f(G, \boldsymbol{z})$. Concretely, its flow f is

$$D = f(G, z) = D(0) + \int_0^t g_\theta(G, D(\tau), z)d\tau, \tag{18}$$

where g_θ is a Message Passing Neural Networks (MPNN) and $D(0) \sim N(0, I)$ is a base distribution. To transfer $p(\boldsymbol{R}|G)$ to a conditional model $p(\boldsymbol{R}|G, \boldsymbol{R}_0)$, we modify the flow to

$$\begin{pmatrix} D_0 \\ D_L \end{pmatrix} = f(G, z, D_0) = \begin{pmatrix} D_0 \\ D_L(0) \end{pmatrix} + \int_0^t \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} g_\theta(G, D_0, D_L(\tau), z)d\tau \tag{19}$$

where $D_0 = \{d_{i,j} = \|\boldsymbol{r}_i - \boldsymbol{r}_j\| | i, j \in G_0\}$ are distances we already knew while $D_L = \{d_{i,j} = \|\boldsymbol{r}_i - \boldsymbol{r}_j\| | i \in G - G_0 \text{ or } j \in G - G_0\}$ are distances between nodes with at least one is in linker. After distance matrix D is predicted, we transform it into absolute coordinates by optimizing the following:

$$\min_{\{\boldsymbol{r}_i | i \in G - G_0\}} \sum_{i,j \in G} (D_{i,j} - \|\boldsymbol{r}_i - \boldsymbol{r}_j\|)^2. \tag{20}$$

Note that we only need to optimize $\{\boldsymbol{r}_i | i \in G - G_0\}$ since $\boldsymbol{R}_0 = \{\boldsymbol{r}_i | i \in G_0\}$ are given. Also there is no need for coordinates alignment since the coordinate system is well-defined by coordinates of fragments $\boldsymbol{R}_0$. Code is provided by https://github.com/MinkaiXu/ConfVAE-ICML21. Note that although ConfVAE can be trained in an end-to-end manner (both equation 19 and 20), it only makes a few improvements (0.01) compared to training the flow alone (see the experiments of its original paper (Xu et al., 2021)). Thus we choose only to train the flow model alone in our experiments. For each graph, we sample one geometry that is optimized by (20) with ten times random initialization and 300 steps of gradient descent.

Gen3D. Gen3D is an autoregressive model using EGNN as node embedder. Following the same logic of modifying GraphAF, we count loss starting from the two fragments. There are two additional modifications we find useful to make: (1) we predict coordinates only relying on distance distribution; (2) we model the distances as Gaussian distribution (regression) instead of discrete distribution on grids (classification) proposed in the original paper;

D. Ablation Study

We conduct ablation study on two aspects: removing equivariant features and coordinates update strategy. We train these three models for 20 epochs and evaluate them by the same methods in previous experiments. Results are shown in Table 4 and 5. We can see a dramatic drop of performances after moving either equivariant features or coordiantes update.

Table 4: Ablation Study. (eqv-) stands for removing equivariant features while (update-) means removing coordinates update strategy.

Metrics	Valid (%)	Recovered (%)	Pass 2D filters (%)	RMSD	Unique (%)	novel (%)
3DLinker	98.67	93.58	90.37	0.079	29.42	32.48
3DLinker (eqv-)	99.42	86.59	92.68	1.352	34.58	27.02
3DLinker (update-)	98.85	39.94	62.81	0.399	55.93	72.25

Table 5: Ablation study. (eqv-) stands for removing equivariant features while (update-) means removing coordinates update strategy.

Metrics	$\mathrm{SC}_{\mathrm{RDKit}}$ Fragments			
	> 0.7 (%)	> 0.8 (%)	> 0.9 (%)	Average
3DLinker	42.55	15.85	2.49	0.683
3DLinker (eqv-)	38.51	13.15	1.76	0.672
3DLinker (update-)	37.34	10.87	1.13	0.670

Especially, both these two modules contribute greatly to the prediction of coordinates, leading to a decrease of RMSD by 1.3 and 0.3 respectively. Also it is interesting to see that coordinates update has a significant impact on graph quality. A possible reason is that updating the coordinates results in a flexible intermediate coordinates, which may increase the expressive capacity of features. In some sense it is similar to EGNN (Satorras et al., 2021b), who also update intermediate coordinates in the forward pass.

E. More Visualizations

For some of the ground-truth fragments, we randomly select 8 generated samples for visualization. As shown in the following figures, each row contains the ground-truth (the most left one) and other 8 generated samples. Note that there are examples showing missing fragments, which indicates the failure of fragment linking (invalid generation).