DiffMol: 3D Structured Molecule Generation with Discrete Denoising Diffusion Probabilistic Models

Weitong Zhang [*†1] **Xiaoyun Wang** [*2] **Justin Smith** [2] **Joe Eaton** [2] **Brad Rees** [2] **Quanquan Gu** [1]

Abstract

3D structures of molecules are often required to investigate atomistic phenomena accurately in industries such as drug design. We propose `DiffMol`, a novel method that utilizes diffusion models to generate the 3D position of atoms and utilizes the discrete denoising diffusion process to generate the atom type. Compared to existing methods, our algorithm offers greater flexibility for post-processing and refining the generated molecules and demonstrates faster performance. We provide theoretical proof of the equivariance of the diffusion process for molecule position generation. Our model achieved better than state-of-the-art performance in molecule/atom stability and molecule validity on benchmarks generating 3D molecules.

1 Introduction

Drug development is a complex and time-consuming process that involves the discovery of small molecules that can effectively bind to target proteins and slow or stop disease progression while remaining non-toxic and not disrupting other biological processes. Generative machine learning models have become an important tool in this process by enabling the generation of new molecules with improved properties and binding affinities. Among the various generative models available, diffusion-based models have emerged as a promising approach, with demonstrated success in image generation and molecule generation. In particular, Equivariant Diffusion Models (EDMs) (Hoogeboom et al., 2022) use Equivariant Graph Neural Networks (EGNN) (Satorras et al., 2021) to generate molecules with 3D structures. Many studies have explored injecting prior knowledge for both the molecule generation process and the binding process (Trippe et al., 2023; Corso et al., 2023; Lee et al., 2023; Igashov et al., 2022).

Existing works usually use the same model to generate the atom types and the atom positions, which consist of discrete features and continuous features. For example, Hoogeboom et al. (2022) encodes the atom type as a one-hot embedding and learns the generation process for these one-hot features. Such a generation process is not quite necessary for discrete features. Inspired by Austin et al. (2021) and other masked language generative models (Devlin et al., 2019; Raffel et al., 2020; Clark et al., 2020), we propose using Discrete Denoising Diffusion Probabilistic Models (D3PMs) to generate the atom type while keeping the continuous diffusion process to generate the atom positions as a separate model. Besides that, we also give a theoretical justification for the equivariance of the generated atom position, i.e., the generation is not affected w.r.t the translation or rotation of the molecules. Our contributions are summarized as follows

- We propose a unified framework to generate 3D molecules where we incorporate the atom-type generation and atom-position generation using the Discrete Denoising Diffusion Probabilistic Models and continuous diffusion model, respectively.
- Our unified framework brings more flexibility to post-process and refine the generated molecules. We demonstrate the effectiveness of generating 3D molecules with state-of-art molecule stability and validity.

Notation. Scalars and constants are denoted by lower and upper case letters, respectively. Vectors are denoted by lower case boldface letters $\mathbf{x}$, and matrices by upper case boldface letters $\mathbf{A}$. We denote by $[k]$ the set $\{1, 2, \cdots, k\}$ for positive integers k. For a discrete random variable $h \in [K]$, we denote the onehot embedding of this random variable as $\mathbf{h} = (0, \cdots, 1, \cdots, 0)^\top$.We further define the categorical distribution of discrete variable h as $\text{Cat}(h, \mathbf{p})$. In particular, $\mathbb{P}(h = i) = \mathbf{p}_i$, where $\mathbf{p}_i$ is the i-th element of the vector $\mathbf{p} \in \mathbb{R}^K$.

[*]Equal contribution [†]Work partially done during an internship at NVIDIA [1]Department of Computer Science, University of California Los Angeles, Los Angeles, CA, USA. [2]NVIDIA, Santa Clara, CA, USA.. Correspondence to: Weitong Zhang <weightzero@ucla.edu>, Xiaoyun Wang <xiaoyunw@nvidia.com>.

Accepted to ICML workshop on Structured Probabilistic Inference & Generative Modeling

2 Preliminaries and Related works

We provide the preliminary knowledge and the most related works in this section, additional related works are provided in Appendix A.

Equivariant models for 3D molecule structures. In a molecule, the 3D position of each atom is denoted as a point cloud $\{\mathbf{x}^n\}_{n=1}^N$, where $\mathbf{x}_n \in \mathbb{R}^3$, and the type of each atom is represented by $\{h^n\}_{n=1}^N$. The global scalar features of the molecules, such as heat capacity and polarizability, are represented by a feature vector $\mathbf{f} \in \mathbb{R}^d$. Obviously, the atom type and the scalar features are *invariant* against the rotation and translation of the molecules, and the spatial features, like the atom positions, are *equivariant* by the aforementioned operations. To formally describe this observation, we say a function $\mathbf{z}_x, \mathbf{z}_h = f(\{\mathbf{x}^n\}_{n=1}^N, \{h^n\}_{n=1}^N, \cdots)$ is $\mathrm{SE}(3)$ equivariant if for any $\mathbf{R} \in \mathrm{SO}(3, \mathbb{R})$ and $\mathbf{t} \in \mathbb{R}^3$ we have

$$\mathbf{R}\mathbf{z}_x, \mathbf{z}_h = f(\{\mathbf{R}\mathbf{x}^n + \mathbf{t}\}_{n=1}^N, \{h_n\}_{n=1}^N, \cdots), \tag{2.1}$$

where the $\cdots$ represents other inputs in the function that remain unchanged. A series of Equivariant Graph Neural Networks (EGNNs) (Fuchs et al., 2020; Satorras et al., 2021; Thölke & Fabritiis, 2022) are proposed to implement a $\mathrm{SE}(3)$ equivariant function defined in (2.1) .

Score-based generative models. We follow Song et al. (2020) to learn the probability distribution of the atom positions $\mathbb{P}(\{\mathbf{x}_n\}_{n=1}^N)$. The forward process is described by the following stochastic differential equations (SDEs)

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}, \tag{2.2}$$

where $\mathbf{w}$ is the standard Brownian motion. The reverse process can be described as the following SDEs:

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log \mathbb{P}_t(\mathbf{x})\right] + g(t)\mathrm{d}\bar{\mathbf{w}}, \tag{2.3}$$

where $\bar{\mathbf{w}}$ is the standard time reversal Brownian motion. The reverse process can also be presented as the following ODEs

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - 0.5g^2(t)\nabla_{\mathbf{x}} \log \mathbb{P}_t(\mathbf{x})\right], \tag{2.4}$$

and it is also known as the 'denoising process' when $t \to 0$ (Song et al., 2021; 2020). When setting $\mathbf{f}(\mathbf{x}, t) = -\frac{1}{2}\beta_t\mathbf{x}, g(t) = \sqrt{\beta_t}$, one can show that $\mathbf{x}_t|\mathbf{x}_0 \sim \mathcal{N}(\mu_t\mathbf{x}_0, \sigma_t^2\mathbf{I})$ where $\mu_t = \exp\left(-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s\right)$ and $\sigma_t^2 = 1 - \exp\left(-\int_0^t \beta(s)\mathrm{d}s\right)$. The score function $\nabla_{\mathbf{x}} \log \mathbb{P}_t(x)$ is approximated by a neural network $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ which is trained by minimizing the objective function

$$\mathcal{L} = \mathbb{E}\left[\lambda_t\left[\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log \mathbb{P}_{0t}(\mathbf{x}_t|\mathbf{x}_0)\|^2\right]\right], \tag{2.5}$$

where λ_t is a weight regularizer. The expectation is taken over $t \sim \mathrm{Unif.}([0, T])$, $\mathbf{x}_0$ is sampled from the initial distribution $\mathbb{P}_0(\mathbf{x})$, $\mathbf{x}_t$ is sampled from the forward process (2.2) given $\mathbf{x}_0$ and $\mathbb{P}_{0t}$ is the corresponding normal distribution. After getting $\mathbf{s}_{\boldsymbol{\theta}}$, the concerned distribution $\mathbb{P}_0(\mathbf{x})$ can be recovered by following the reversed process (2.3).

Discrete Denoising Diffusion Probabilistic Models. We follow Austin et al. (2021) to handle the discrete distribution of the atom types $\mathbb{P}(\{h_n\}_{n=1}^N)$. For the scalar discrete random variables with K categories, the forward transition is defined by the transition matrix $\mathbf{Q}_t \in \mathbb{R}^{K\times K}$. Denoting the one-hot embedding of the discrete random variable h as $\mathbf{h} \in \mathbb{R}^{1\times K}$, then the one-step forward process is defined by $\mathbb{P}(h_t|h_{t-1}) = \mathrm{Cat}(h_t, \mathbf{p} = \mathbf{h}_{t-1}\mathbf{Q}_t)$. Thus the t-step marginal distribution given h_0 is defined by

$$\mathbb{P}(h_t|h_0) = \mathrm{Cat}\left(h_t, \mathbf{p} = \mathbf{h}_0 \textstyle\prod_{\iota=1}^t \mathbf{Q}_\iota\right), \tag{2.6}$$

where $\prod_{\iota=1}^t \mathbf{Q}_\iota := \mathbf{Q}_1\mathbf{Q}_2\cdots\mathbf{Q}_t$. The reverse process is sampled by $h_T \sim \mathbb{P}_T(\cdot)$ then for all $t = T, T-1, \cdots, 1$.

$$h_{t,0} \sim \mathbb{P}_{0|t}(\cdot|h_t); h_{t-1} \sim \mathbb{P}_t(\cdot|h_t, h_{t,0}). \tag{2.7}$$

$\mathbb{P}_{0|t}$ denotes the posterior distribution of h_0 given h_t and the $\mathbb{P}_t$ denotes the posterior distribution of h_{t-1} given h_0 and h_t. It's easy to show that once we have a good estimation of $\mathbb{P}_{t,0}$ and $\mathbb{P}_t$, $h_{t-1} \sim \mathbb{P}_{t-1|t}(\cdot|h_t)$ is exactly the posterior distribution of h_{t-1} conditioned on h_t.

Usually, $\mathbb{P}_t(\cdot|h_t)$ can be readily derived once we know the transition kernel $\mathbf{Q}$. Thus we approximate the $\mathbb{P}_{0|t}$ with a neural network $\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}$. As discussed in Austin et al. (2021), $\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}$ can be trained by minimizing the following negative-log-likelihood function

$$\mathcal{L}_h = -\mathbb{E}_{h_0}\left[\textstyle\sum_{t=0}^T \mathbb{E}_{h_t|\mathbf{h}_0}\left[\log \widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}(h_0|h_t)\right]\right], \tag{2.8}$$

where the expectation is taken over the initial distribution of h_0 and the h_t given the forward process (2.6).

3 Problem Definition

We consider the 3D-structured molecules with N atoms and atoms index up to K, where the position of the atoms are represented as a point cloud $\{\mathbf{x}^n\}_{n=1}^N$ and the type of atoms are represented by $\{h^n\}_{n=1}^N$. We aim to learn the score function $\{\mathbf{s}_{\boldsymbol{\theta},t}^n\}$ and the distribution $\{\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}^n\}$ for all atoms $n \in [N]$ so that we can follow (2.3) and (2.7) to generate the molecule structure $\{\mathbf{x}_0^n, h_0^n\}$. We use $\mathbf{x}_t^n$ to denote the position of n-th atom at time t and use h_t^n to denote the type of n-th atom at time t. When applying the generative model to atom positions, our objective is to maintain the independence of molecule structure generation from molecule translation.

Assumption 3.1. Denote $\mathbf{x}_t^c = \frac{1}{N}\sum_{n=1}^N \mathbf{x}_t^n$ as the geometric center of a molecule, we assume the probability distribution of the molecule structure $\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N$ is independent of the probability distribution of the massive center $\mathbf{x}_t^c$. In other words, we have for all $t \in [T]$,

$$\mathbb{P}_t(\{\mathbf{x}_t^n\}_{n=1}^N) = \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N)\mathbb{P}_t(\mathbf{x}_t^c). \tag{3.1}$$

We also assume the probability $\mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N)$ is invariant against the rotation $\mathbf{Q} \in \mathrm{SO}(3)$, i.e. for all $\mathbf{Q} \in \mathrm{SO}(3)$, $\mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N) = \mathbb{P}_t(\{\mathbf{Q}(\mathbf{x}_t^n - \mathbf{x}_t^c)\}_{n=1}^N)$.

Under Assumption 3.1, it's easy to show that for any $\mathbf{x}_t^n$, we have

$$\begin{aligned}
&\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N) \\
=&\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n\}_{n=1}^N) - \nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\mathbf{x}_t^c). \\
&\mathbf{Q}\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N) \\
=&\nabla_{\mathbf{Q}\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{Q}\mathbf{x}_t^n - \mathbf{Q}\mathbf{x}_t^c\}_{n=1}^N),
\end{aligned} \tag{3.2}$$

therefore we have that the score $\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N)$ is translation invariant and rotation equivariant, thus the score function and the distribution $\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}^n$ can be approximated by an EGNN

$$\{\mathbf{s}_{\boldsymbol{\theta},t}^n(\cdot)\}, \{\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}^n(\cdot)\} = \mathrm{EGNN}_{\boldsymbol{\theta}}(\{\mathbf{x}_t^n\}, \{h_t^n\}, t), \tag{3.3}$$

where we use $\cdot$ to represent the formal parameter $\{\mathbf{x}_t^n\}$, $\{h_t^n\}$ and t which is the same with the input of EGNN.

4 Methodology

4.1 Equivariant diffusion model for position generation

Following (2.5), the loss function is defined by

$$\mathcal{L}_1 = \mathbb{E}\left[\lambda_t \sum_{n=1}^N \left\|\mathbf{s}_{\boldsymbol{\theta},t}^n(\cdot) - \nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\})\right\|_2\right], \tag{4.1}$$

where the expectation is taken over $t \in \mathrm{Unif}(t)$, $\{\mathbf{x}_0^n\}$ following the initial distribution and $\{\mathbf{x}_t^n\}$ following the forward diffusion process described in (2.2) for each $n \in [N]$. The score function of $\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\})$ is defined by (3.2) and the following lemma shows that the decomposition in (3.2) is similar with the zero-centered denoising process in Hoogeboom et al. (2022) for estimating (4.1).

Lemma 4.1. Denote $\boldsymbol{\epsilon}_t^n = \mathbf{x}_t^n - \mu_t \mathbf{x}_0^n \sim \mathcal{N}(\mathbf{0}, \sigma_t^2 \mathbf{I})$. Then for any $(n,t) \in [N] \times [0,T]$, we have that

$$\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}) = -\frac{\boldsymbol{\epsilon}_t^n}{\sigma_t^2} + \sum_{n=1}^N \frac{\boldsymbol{\epsilon}_{n,t}}{N\sigma_t^2}. \tag{4.2}$$

After learning the score function of $\mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\})$, under Assumption 3.1, a new distribution of $\mathbb{P}_0'(\{\mathbf{x}_{n,0}\})$ can be reconstructed by letting $\mathbb{P}_t'(\{\mathbf{x}_t^n\}) = \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\})\mathbb{P}_0'(\mathbf{x}_t^c)$ where $\mathbb{P}_t'(\mathbf{x}_t^c) = \mathcal{N}(\mathbf{0}, \sigma_{c,t}^2 \mathbf{I})$. Thus the reverse process of each atom $n \in [N]$ is

$$\mathrm{d}\mathbf{x}_t^n \approx \left[-0.5\beta_t \mathbf{x}_t^n - \beta_t^2 \widetilde{\mathbf{s}}_{\boldsymbol{\theta},n}(\cdot) - \frac{\beta_t^2 \mathbf{x}_t^c}{N\sigma_{c,t}^2}\right]\mathrm{d}t + \sqrt{\beta_t}\mathrm{d}\bar{\mathbf{w}}, \tag{4.3}$$

where term $\beta_t^2 \mathbf{x}_t^c / (N\sigma_{c,t}^2)$ corresponds to the translation of the point cloud. Since $\widetilde{\mathbf{s}}(\cdot)$ is translation-invariant and the absolute position of the generated molecule is irrelevant, we can safely drop term $\beta_t^2 \mathbf{x}_t^c / (N\sigma_{c,t}^2)$ during the generation process. The detailed proof of Lemma 4.1 and the reverse process (4.3) are deferred into Appendix B.

4.2 Structured diffusion process for atom-type generation

In this subsection, we learn the model for atom-type generation. We denote h_t^n and $\mathbf{h}_{n,t}$ as the atom type and the one-hot embedding for n-th time at time $t \in [T]$. Following the setting of *absorbing states* in Austin et al. (2021), each atom can be randomly turned into an absorbing state during the diffusion process. We denote the absorbing state as "#". Then the transition kernel for any $(n,t) \in [N] \times [T]$ is described as follows:

$$\begin{cases}
\mathbb{P}_t(h_t^n = \#|h_{n,t-1} = \#) = 1 \\
\mathbb{P}_t(h_t^n = h_{n,t-1}|h_{n,t-1} \neq \#) = \frac{T-t}{T-t+1} \\
\mathbb{P}_t(h_t^n = \#|h_{n,t-1} \neq \#) = \frac{1}{T-t+1} \\
\mathbb{P}_t(h_t^n|h_{n,t-1}) = 0
\end{cases} \tag{4.4}$$

This transition kernel also corresponds to the one used in Generative Masked Language Models (Chen et al., 2023; Li et al., 2022) and it's easy to prove that $\mathbb{P}_{0t}(h_t^n = \#|h_0^n) = t/T$ which means all atoms will be transited to absorbing state # at time $t = T$. Based on the loss (2.8), $\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}^n(\cdot|\cdots)$ is learnt by minimizing the following negative-log-likelihood over all atoms $m \in [N]$:

$$\mathcal{L}_2 = \mathbb{E}\left[\sum_{m=1}^N \log \widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}^m(h_0^m|\{h_t^n\}, \cdots)\right], \tag{4.5}$$

where the expectation is taken from $t \sim \mathrm{Unif}.[T]$, $\{h_0^n\}$ from the initial distribution and $\{h_t^n\}$ is generated by the diffusion process given by (2.6).

Given $\{\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}^n\}(\cdot)$ and the transition kernel (4.4), the reverse process (2.7) starts from initializing $h_T^n = \#$ for all $n \in [N]$. Then for $t = T, T-1, \cdots, 1$, we first sample $h_{t,0}^n$ from $\{\widetilde{\mathbb{P}}_{\boldsymbol{\theta},t}^n\}(\cdot)$ for each $n \in [N]$. After that, for each $n \in [N]$, independently we sampled h_{t-1}^n following

$$\begin{cases}
\mathbb{P}_t(h_{t-1}^n = \#|h_t^n = \#, h_{0,t}^n) = 1 - \frac{1}{t} \\
\mathbb{P}_t(h_{t-1}^n = h_{0,t}^n|h_t^n = \#, h_{0,t}^n) = \frac{1}{t} \\
\mathbb{P}_t(h_{t-1}^n = h_t^n|h_t^n \neq \#, h_{0,t}^n) = 1 \\
\mathbb{P}_t(h_t^n|h_{t-1}^n, h_{0,t}^n) = 0 \text{ otherwise}
\end{cases}, \tag{4.6}$$

Algorithm 1 Generative model training

Input: Molecule position $\{\mathbf{x}_0^n\}$ and type $\{h_0^n\}$.
1: Sample $t \in \text{Unif.}([T])$
2: Generate perturbation $\mathbf{x}_{n,t} \sim \mathcal{N}(\mu_t \mathbf{x}_{n,0}, \sigma_t^2 \mathbf{I})$ for all $n \in [N]$.
3: W.p. t/T, set $h_{n,t} = \#$, otherwise set $h_{n,t} = h_{n,0}$
4: Set $\{\mathbf{s}^n_{\boldsymbol{\theta},t}(\cdot)\}, \{\widetilde{\mathbb{P}}^n_{\boldsymbol{\theta},t}(\cdot)\} = \text{EGNN}_{\boldsymbol{\theta}}(\{\mathbf{x}_t^n\}, \{h_t^n\}, t)$
5: Optimize $\boldsymbol{\theta}$ by minimizing $\mathcal{L}_1(\boldsymbol{\theta}) + \mathcal{L}_2(\boldsymbol{\theta})$
Output: Return network $\text{EGNN}(\boldsymbol{\theta})$.

Algorithm 2 3D structured molecule generation

Input: Number of atom N.
1: Initialize $\mathbf{x}_T^n \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $h_T^n = \#$ for all $n \in [N]$.
2: **for** $t = T, \cdots, 1$ **do**
3: Set $\{\mathbf{s}^n_{\boldsymbol{\theta},t}(\cdot)\}, \{\widetilde{\mathbb{P}}^n_{\boldsymbol{\theta},t}(\cdot)\} = \text{EGNN}_{\boldsymbol{\theta}}(\{\mathbf{x}_t^n\}, \{h_t^n\}, t)$
4: **for** $n \in [N]$ **do**
5: Sample $\boldsymbol{\epsilon}_t^n \sim \mathcal{N}(0, \beta_t/T)$
6: Let $\mathbf{x}_{t-1}^n = \mathbf{x}_t^n + [0.5\beta_t \mathbf{x}_t^n + \beta_t^2 \widetilde{\mathbf{s}}_{\boldsymbol{\theta},n}(\cdot)]/T + \boldsymbol{\epsilon}_t^n$
7: Sample $h^n_{0,t}$ from $\widetilde{\mathbb{P}}^n_{\boldsymbol{\theta},t}(\cdot)$
8: Sample h_{t-1}^n following (4.6).
9: **end for**
10: **end for**
Output: Return $\{\mathbf{x}_0^n\}_{n=1}^N, \{h_0^n\}_{n=1}^N$

In simple terms, we take the sampled initial state $h^n_{0,t}$ with a gradually increasing probability as t approaches 1. Detailed proof of this reverse transition kernel can be found in Appendix B. Once h_{t-1}^n is assigned with a non-masked token, it will not change during the generation process.

4.3 Proposed algorithms

Combining the score-based model for position generation and the structured diffusion process for atom-type generation, the training algorithm and the generation algorithm are presented in Algorithm 1 and Algorithm 2. In particular, we use Equivariant-Transformer (Thölke & Fabritiis, 2022) as our backbone network and use the Euler method with step size $1/T$ to solve the reverse SDE (4.3) numerically and we drop the term regarding the translation of the geometric center $\mathbf{x}_t^c$ in (4.3) since we do not care the absolute position of the generated molecule.

4.4 Post-Generation Refinement

Besides the main algorithm for training and generating the molecules, the following patches proved beneficial for generating high-quality molecules.

Denoising process with $t \to 0$. As previously mentioned, the diffusion process with $t \to 0$ can be viewed as a denoising process. In particular, the score-based generative model using (2.4) can be viewed as canceling the perturbation of the position, and the structured diffusion process using (2.7) can be viewed as correcting the atom type. Thus, we can refine the generated molecule further by iterating with $t = 1$ for an additional T' times using Algorithm 2.

Bond construction. After getting the position $\{\mathbf{x}^n\}$ and atom type $\{h^n\}$, instead of using the distance between every two atoms to determine the bond type, we use `xyz2mol` provided in Rdkit (Landrum et al., 2016) to construct the bond between atoms. This eliminates the need for bond margin hyperparameters in Hoogeboom et al. (2022).

After generating the molecule structure, we can neutralize any charged atom by changing the atom type with the correct element to attain an octet in its valence shell. For example, if there is only a double bond connected to a nitrogen atom, we can replace it with an oxygen atom thus neutralizing the molecule. It's true that the molecule structure might be changed by replacing some of the atoms. However, since the atom type generation is based on the structured discrete denoising diffusion probabilistic models, we can always correct this by running Algorithm 2 to refine the structure.

5 Experiments

Unconditioned Molecule Generation. We trained our molecule generative model using the QM9 dataset (Ramakrishnan et al., 2014) as our benchmark in this paper. The QM9 dataset comprises calculated properties of 134k organic molecules containing four kinds of heavy atoms (C, N, O, F). We followed the train/validation/test split used in (Hoogeboom et al., 2022), which consists of 100k/18k/13k samples respectively for each partition.

The result of QM9 is presented in Table 1. We use the same metric as provided in Hoogeboom et al. (2022) and other related works. Here **Atom Stability** means the proportion of atoms that have the right valency and **Molecule Stability** means the proportion of molecules where all atoms are stable. **Validity** means the proportion of molecules having a valid SMILES representation. As the result suggests, due to our structured discrete denoising diffusion probabilistic models for atom types and the neutralization post-processing, our model enjoys higher stability for both atoms and molecules. For more experimental results, please refer to C.1

Table 1. Selected result on unconditioned molecule generation trained on QM9. **Validity**, **Atom Stability** and **Molecule Stability** represent the percentage (%) of these metrics.

Models	Validity	Atom Stability	Molecule Stability
G-Schnet	85.5	95.7	68.1
EDM	91.9	98.7	82.0
DiffMol(Ours)	**98.03**	**99.62**	**95.41**
QM9 Data	97.7	99.0	95.2

We also conducted several ablation studies regarding the performance impact of each component in our model. We

defer this discussion to Appendix C.3 due to page limits.

Conditional Generation. We conduct the conditional generation following the same setting as Hoogeboom et al. (2022). Due to the page limit the details are deferred to Appendix C.2

6 Conclusion

In conclusion, we have proposed a novel approach for 3D molecule generation through our DiffMol Model, which utilizes an equivariant diffusion model and splits the diffusion process for atom types and positions. By incorporating the neutralization check provided by RdKit, we were able to achieve exceptional performance that surpassed previous state-of-the-art results. Our model's ability to generate diverse and high-quality molecules with accurate atom placements makes it a promising tool for drug discovery and material science. Additionally, the unique combination of our approach with the neutralization check provides an added level of confidence in the validity and utility of the generated molecules. Future work could explore further enhancements to our model and extend its applicability to protein applications.

Acknowledgements

We thank the anonymous reviewers for their helpful comments.

References

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021.

Chen, T., ZHANG, R., and Hinton, G. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3itjR9QxFw.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. Electra: Pre-training text encoders as discriminators rather than generators, 2020.

Corso, G., Stärk, H., Jing, B., Barzilay, R., and Jaakkola, T. Diffdock: Diffusion steps, twists, and turns for molecular docking, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

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

Gasteiger, J., Giri, S., Margraf, J. T., and Günnemann, S. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. In *Machine Learning for Molecules Workshop, NeurIPS*, 2020.

Hoogeboom, E., Satorras, V. G., Vignac, C., and Welling, M. Equivariant diffusion for molecule generation in 3d. In *International Conference on Machine Learning*, pp. 8867–8887. PMLR, 2022.

Igashov, I., Stärk, H., Vignac, C., Satorras, V. G., Frossard, P., Welling, M., Bronstein, M., and Correia, B. Equivariant 3d-conditional diffusion models for molecular linker design, 2022.

Jing, B., Corso, G., Chang, J., Barzilay, R., and Jaakkola, T. Torsional diffusion for molecular conformer generation. *ArXiv*, abs/2206.01729, 2022.

Landrum, G. et al. Rdkit: Open-source cheminformatics software. 2016.

Lee, J. S., Kim, J., and Kim, P. M. Proteinsgm: Score-based generative modeling for de novo protein design. *bioRxiv*, 2023.

Li, X. L., Thickstun, J., Gulrajani, I., Liang, P., and Hashimoto, T. B. Diffusion-lm improves controllable text generation, 2022.

Lin, H., Huang, Y., Liu, M., Li, X. C., Ji, S., and Li, S. Z. Diffbp: Generative diffusion of 3d molecules for target protein binding. *ArXiv*, abs/2211.11214, 2022.

Liu, M., Luo, Y., Uchino, K., Maruhashi, K., and Ji, S. Generating 3d molecules for target protein binding. *ArXiv*, abs/2204.09410, 2022.

Lu, S., Yao, L., Chen, X., Zheng, H., and Ke, G. 3d molecular generation by virtual dynamics, 2023. URL https://openreview.net/forum?id=tZmqS73_07.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

Ramakrishnan, R., Dral, P. O., Rupp, M., and von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014.

Satorras, V. G., Hoogeboom, E., and Welling, M. E(n) equivariant graph neural networks. *CoRR*, abs/2102.09844, 2021. URL https://arxiv.org/abs/2102.09844.

Satorras, V. G., Hoogeboom, E., Fuchs, F. B., Posner, I., and Welling, M. E(n) equivariant normalizing flows, 2022.

Schneuing, A., Du, Y., Harris, C., Jamasb, A., Igashov, I., Du, W., Blundell, T., Lió, P., Gomes, C., Welling, M., Bronstein, M., and Correia, B. Structure-based drug design with equivariant diffusion models, 2022.

Schütt, K. T., Unke, O. T., and Gastegger, M. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *International Conference on Machine Learning*, 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*, 2020. URL https://openreview.net/forum?id=S1esMkHYPr.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations, 2021.

Thölke, P. and Fabritiis, G. D. Torchmd-net: Equivariant transformers for neural network based molecular potentials. *CoRR*, abs/2202.02541, 2022. URL https://arxiv.org/abs/2202.02541.

Trippe, B. L., Yim, J., Tischer, D., Baker, D., Broderick, T., Barzilay, R., and Jaakkola, T. Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem, 2023.

Vignac, C., Osman, N., Toni, L., and Frossard, P. Midi: Mixed graph and 3d denoising diffusion for molecule generation. *ArXiv*, abs/2302.09048, 2023.

Wu, L., Gong, C., Liu, X., Ye, M., and Liu, Q. Diffusion-based molecule generation with informative prior bridges. *ArXiv*, abs/2209.00865, 2022.

Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., and Tang, J. Geodiff: A geometric diffusion model for molecular conformation generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=PzcvxEMzvQC.

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

A Additional Related Work

In previous years, the generation models are sequential models. Zang & Wang (2020) proposed a flow-based graph generative model to learn invertible mappings between molecular graphs and their latent representations, MoFlow (Zang & Wang, 2020) generates bonds (edges) through a Glow-based model followed by generating atoms (nodes) with a final posthoc validity correction. Shi* et al. (2020) proposed a flow-based autoregressive model for graph generation method to generate molecules. However, these flow-based based methods can only generate graph structures and cannot generate 3D structured molecules, and the sequential models are not time efficient.

With the development of molecular GNNs, a group of equivariant algorithms has come out. Fuchs et al. (2020); Schütt et al. (2021); Thölke & Fabritiis (2022); Satorras et al. (2021); Gasteiger et al. (2020) In the Diffusion models, the invariant property is used from the molecular GNN models (Hoogeboom et al., 2022).

In recent years, diffusion models have been playing an essential role in molecular generation tasks. Diffusion models have been used on molecular conformer generation in computational chemistry; GeoDiff (Xu et al., 2022) treats each atom as a particle and learns to reverse the diffusion process as a Markov chain, with equivariance property. Jing et al. (2022) operates on the space of torsion angles via a diffusion process on the hypertorus and an extrinsic-to-intrinsic score model. Satorras et al. (2022) is the first flow that jointly generates molecules features and positions in 3D with E(n) equivariant normalizing flows. Hoogeboom et al. (2022) an equivariant network to deal with the movement such as rotation in the molecules with allowance for the computation of likelihoods for molecules. Wu et al. (2022) proposed a method that incorporates physical and statistical information to improve the training of diffusion-based generative models. The proposed method employs a Lyapunov function to construct and determine these bridges, and offers several informative prior bridges for molecule generation. A mixture of a graph neural network Satorras et al. (2021) and diffusion model also works on molecule generation (Vignac et al., 2023), which jointly generates molecular graphs and corresponding 3D conformers.

Diffusion models are also useful a lot in drug discovery tasks such as target protein binding. Schneuing et al. (2022) proposes a E(3)-equivariant 3D-conditional diffusion model to generate ligands on protein pockets. Lin et al. (2022) proposes a non-autoregressive generative diffusion model for molecular 3D structures, which utilizes target proteins as contextual constraints at a full-atom level. The model generates both element types and 3D coordinates of the molecule using an equivariant network. Liu et al. (2022) uses a graph neural network to obtain informative representations from contextual information, then a local reference atom is selected and a local coordinate system is constructed to place a new atom via a flow model. Variables are generated sequentially to capture dependencies. Lu et al. (2023) proposed an end-to-end model generating fine-grained 3D molecules with binding positions in the pocket cavity.

Computer vision methods are also used in the molecule generation for protein design, a score-based diffusion model (Lee et al., 2023) generates de novo proteins using image-based representations of protein structure, and also inpaints plausible backbones and domains into structures of predefined length.

B Proof of Concepts

We provide the proof of concepts for the diffusion processes discussed in Section 4 in this section.

Proof of Lemma 4.1. According to (3.2) and the fact that $\mathbf{x}_t^n|\mathbf{x}_0^n \sim \mathcal{N}(\mu_t\mathbf{x}_0^n, \sigma_t^2\mathbf{I})$, we have that

$$\begin{aligned}
\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N) &= \nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n\}_{n=1}^N) - \nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\mathbf{x}_t^c) \\
&= \nabla_{\mathbf{x}_t^n} \left(-\frac{\|\mathbf{x}_t^n - \mu_t\mathbf{x}_0^n\|_2^2}{2\sigma_t^2} \right) - \nabla_{\mathbf{x}_t^n} \left(-\frac{\|\mathbf{x}_t^c - \mu_t\mathbf{x}_0^c\|_2^2}{2\sigma_t^2/N} \right) \\
&= -\frac{\mathbf{x}_t^n - \mu_t\mathbf{x}_0^n}{\sigma_t^2} - \frac{\mathbf{x}_t^n - \mu_t\mathbf{x}_0^n}{\sigma_t^2/N} \cdot \frac{1}{N} \\
&= -\frac{\boldsymbol{\epsilon}_t^n}{\sigma_t^2} + \sum_{n=1}^N \frac{\boldsymbol{\epsilon}_t^n}{N\sigma_t^2}
\end{aligned}$$

where the second equation comes from the fact that $\mathbf{x}_t^c|\mathbf{x}_0^c \sim \mathcal{N}(\mu_t\mathbf{x}_0^c, \sigma_t^2\mathbf{I}/N)$ since $\mathbf{x}_t^c$ is the geometric center of $\{\mathbf{x}_t^n\}_n$, the third equation holds due to the fact that $\mathbf{x}_t^c = \sum_n \mathbf{x}_t^n/N$. Then we reach the claimed results by recalling $\boldsymbol{\epsilon}_t^n := \mathbf{x}_t^n - \mu_t\mathbf{x}_0^n$. □

Proof of reverse process (4.3). From the decomposition (3.2), we can have that $\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n\}_{n=1}^N) = \nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n -$

$\mathbf{x}_t^c\}_{n=1}^N) + \nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\mathbf{x}_t^c)$, we only need to provide $\nabla_{\mathbf{x}_t^n} \log \mathbb{P}'_t(\mathbf{x}_t^c)$ while using $\widetilde{\mathbf{s}}_{\boldsymbol{\theta},n}(\cdot)$ to approximate the score function $\nabla_{\mathbf{x}_t^n} \log \mathbb{P}_t(\{\mathbf{x}_t^n - \mathbf{x}_t^c\}_{n=1}^N)$. Since $\mathbf{x}_t^c \sim \mathcal{N}(\mathbf{0}, \sigma_{c,t}^2 \mathbf{I})$, immediately we have

$$\nabla_{\mathbf{x}_t^n} \log \mathbb{P}'_t(\mathbf{x}_t^c) = \nabla_{\mathbf{x}_t^n} \left(-\frac{\|\mathbf{x}_t^c\|_2}{2\sigma_{c,t}^2} \right) = \frac{\mathbf{x}_t^c}{N\sigma_{c,t}^2},$$

where we use the fact that $\mathbf{x}_t^c = \sum_n \mathbf{x}_t^n / N$. One can find that this term remains constant w.r.t different atom $n \in [N]$. Therefore it describe the translation of the whole molecule and we can safely drop that in the algorithm. □

Proof of the discrete reverse process (4.6). It's trivial to show that $\mathbb{P}_t(h_{t-1}^n = h_t^n | h_t^n \neq \#, h_{0,t}^n)$ if h_t^n is not masked, then h_{t-1}^n must have the same token as h_t^n. It then remains to show $\mathbb{P}_t(h_{t-1}^n = h_{0,t}^n | h_t^n = \#, h_{0,t}^n)$. From the forward process (4.4) we have that

$$\begin{aligned}
\mathbb{P}_t(h_{t-1}^n = h_{0,t}^n | h_t^n = \#, h_{0,t}^n) &= \frac{\mathbb{P}_t(h_{t-1}^n = h_{0,t}^n, h_t^n = \# | h_{0,t}^n)}{\mathbb{P}_t(h_t^n = \# | h_{0,t}^n)} \\
&= \frac{\mathbb{P}_t(h_{t-1}^n = h_{0,t}^n | h_{0,t}^n) \cdot \mathbb{P}_t(h_t^n = \# | h_{t-1}^n = h_{0,t}^n, h_{0,t}^n)}{\mathbb{P}_t(h_t^n = \# | h_{0,t}^n)} \\
&= \frac{\left(1 - \frac{t-1}{T}\right) \cdot \left(\frac{1}{T-t+1}\right)}{\frac{t}{T}} \\
&= \frac{1}{t},
\end{aligned}$$

where the second equation comes from the Markovian assumption of the transition kernel and the third equation utilizes the fact from (4.4). Immediately we have that $\mathbb{P}_t(h_{t-1}^n = \# | h_t^n = \#, h_{0,t}^n) = 1 - 1/t$ since there is no other possible token for h_{t-1}^n given the initial token h_0^n. □

C Additional Experiment Results

C.1 Comparison with other experiments

We provide more comparison with other baseline models as in table 2. In addition, we provide a collection of samples of the generated molecules in Figure 1

Table 2. Unconditioned molecule generation trained on QM9. **Validity**, **Atom Stability** and **Molecule Stability** represent the percentage (%) of these metrics.

Models	Validity	Atom Stability	Molecule Stability
E-NF	40.2	85.0	4.9
G-Schnet	85.5	95.7	68.1
GDM-aug	90.4	97.6	71.6
EDM(onehot)	91.9	95.7	46.9
EDM	91.9	98.7	82.0
DiffMol(Ours)	**98.03**	**99.62**	**95.41**
QM9 Data	97.7	99.0	95.2

C.2 Conditional Generation

We conducted conditional generation experiments with varying values of polarizability, denoted as α, within the range of $[60, 120]$. The results of these experiments, as shown in Table 3, indicate that our algorithm exhibits superior validity, atom stability, and molecule stability compared to baseline algorithms (Hoogeboom et al., 2022). Figure 2 presents selected samples of the generated molecules obtained with different α values.

Polarizability is a measure of a molecule's tendency to acquire an electric dipole moment in response to an external electric field. As α increases, we can expect the generation of molecules with less isometric shapes. This behavior is evident in the molecules depicted in Figure 2.

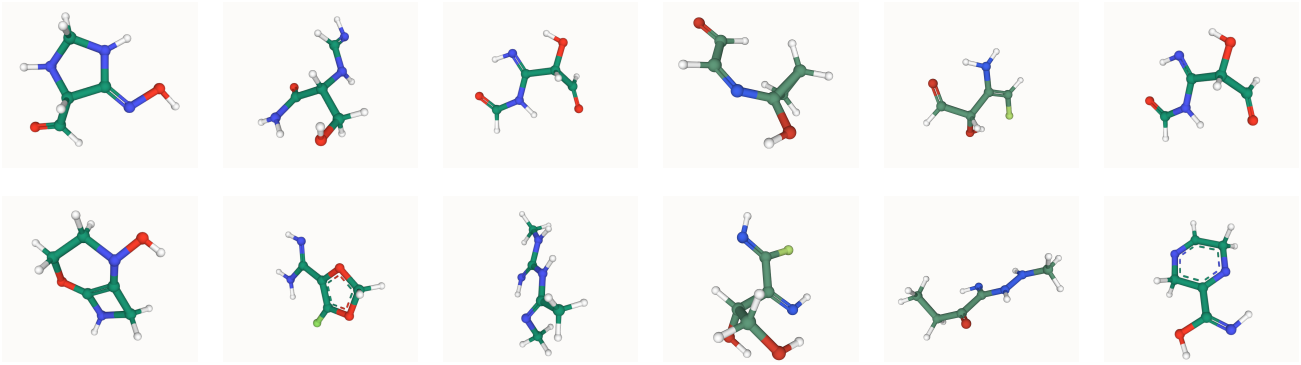

Figure 1. Some samples of unconditioned generated molecules

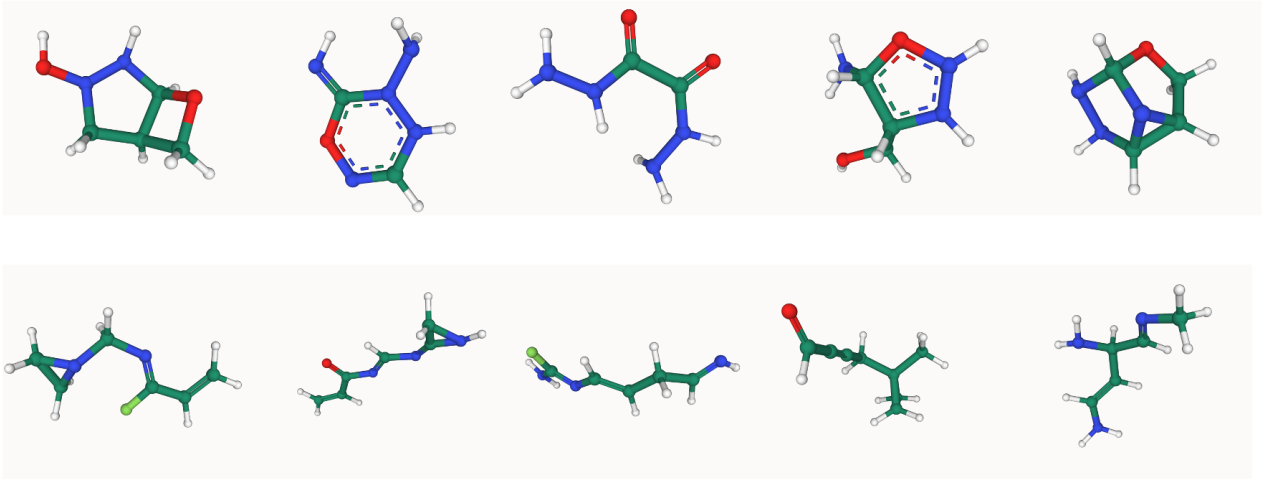

Figure 2. generated samples with different Polarizability values $\alpha = 60$ (top) and $\alpha = 120$ (bottom).

C.3 Ablation Study

We conduct the ablation study for each component of our algorithms.

Using Discrete Denoising Diffusion Probabilistic Models instead of Continuous Diffusion Models..

We would like to discuss if using the D3PMs can improve the quality of the molecule generation. Thus we change the first layer of the network from token embedding (`nn.Embedding()`) to a linear layer (`nn.Linear()`) and use the one-hot embedding for atom-type. The rest pipeline are the same as described previously.

As Table 4 suggests, using DiffMol with one-hot embedding and continuous generation process can achieve a similar molecule stability with EDM (Hoogeboom et al., 2022). However, using D3PMs as we suggested can have better stability, since during the diffusion process, using a discrete token can provide more sparse features compared with the one-hot embedding.

Regarding the length of the diffusion process and the denoising process.

We would like to discuss the impact of the length T to the generation process. In particular, we compare the reverse process with $T = 1000$ and $T = 100$. As Table 5 suggests, our algorithm can still maintain the superior performance given with a shorter $T = 100$. This allows us to accelerate the molecule generation process.

Regarding the quality of the atom neutralization process and xyz2mol bond construction.

Table 3. Conditional Generation trained on QM9 dataset with different Polarizability value α

Models	Validity	Atom Stability	Molecule Stability
GDM	-	75.0	-
GDM-aug	90.4	77.7	-
EDM	91.9	81.3	-
DiffMol (Ours)	**97.60**	**97.59**	**84.45**

Table 4. Ablation study on comparing the one-hot embedding and discrete tokens trained on QM9. **Validity**, **Atom Stability** and **Molecule Stability** represent the percentage (%) of these metrics.

Models	Validity	Atom Stability	Molecule Stability
EDM	91.9	98.7	82.0
DiffMol w/ one-hot embedding	92.5	96.7	81.6
DiffMol w/ discrete (Ours)	**98.03**	**99.62**	**95.41**
QM9 Data	97.7	99.0	95.2

Table 5. Ablation study on comparing impact of time T on the generation quality. **Validity**, **Atom Stability** and **Molecule Stability** represent the percentage (%) of these metrics.

Models	Validity	Atom Stability	Molecule Stability
EDM, $T = 1000$	91.9	98.7	82.0
DiffMol, $T = 100$	97.6	98.7	90.6
DiffMol, $T = 1000$	**98.03**	**99.62**	**95.41**
QM9 Data	97.7	99.0	95.2

Since we do not use the bond construction method provided in Hoogeboom et al. (2022), it worth comparing our generated bonds length with the typical bond length[1]. We report the mean absolute error (MAE) of between the generated bonds and typical bond length. Table 6 to Table 8 provide the MAE for single, double, triple bonds after the neutralization process. Table 9 to Table 11 provide the MAE for single, double, triple bonds after final denoising process.

Several observations can be made from the results provided in these tables. First, the bond length of the generated molecules is close to the typical length. This error is compatible with the margin used in Hoogeboom et al. (2022)[2]. Second, after the neutralization by modifying some of the atoms, doing a denoising process will refine the precision in terms of bond length. Third, there exists some bonds which cannot be well predicted, even with the denoising process (e.g. O−F, N−F, N=N, O=O). We doubt that this is because these bonds are rare in the QM9 dataset so that the diffusion model cannot learn the bond length well.

[1]Following Hoogeboom et al. (2022), the typical bond length can be found at http://chemistry-reference.com/tables/Bond%20Lengths%20and%20Enthalpies.pdf and http://www.wiredchemist.com/chemistry/data/bond_energies_lengths.html

[2]In Hoogeboom et al. (2022), they use a error threshold $m_1 = 10\text{pm}$ for single bonds and $m_2 = 5\text{pm}, m_3 = 3\text{pm}$ for double or triple bonds

Table 6. MAE of single bonds before denoising process (pm)

	H	C	N	O	F
H	-[3]	1.13	2.74	2.85	-
C	1.13	3.70	4.86	4.21	13.23
N	2.74	4.86	10.87	8.31	12.89
O	2.85	4.21	8.31	10.51	17.23
F	-[3]	13.23	12.89	17.23	-[3]

Table 7. MAE of double bonds before denoising process (pm)

	C	N	O
C	4.93	2.73	2.05
N	2.73	2.47	-[3]
O	2.05	-[3]	-[3]

Table 8. MAE of triple bonds before denoising process (pm)

	C	N
C	1.40	2.07
N	2.07	-[3]

Table 9. MAE of single bonds after denoising process (pm)

	H	C	N	O	F
H	-[3]	1.07	2.47	2.76	-[3]
C	1.07	3.47	4.91	3.70	3.81
N	2.47	4.91	8.78	6.54	14.65
O	2.76	3.70	6.54	9.01	32.42
F	-[3]	3.81	14.65	32.42	-[3]

Table 10. MAE of double bonds after denoising process (pm)

	C	N	O
C	3.17	2.63	1.95
N	2.63	6.80	-[3]
O	1.95	-[3]	-[3]

Table 11. MAE of triple bonds after denoising process (pm)

	C	N
C	1.57	1.49
N	1.49	-[3]

[3]We use '-' to denote the bonds which is never appears in the generated molecules.