ChemSpacE: Toward Steerable and Interpretable Chemical Space Exploration

Yuanqi Du
Cornell University
yd392@cornell.edu

Xian Liu
CUHK
alvinliu@ie.cuhk.edu.hk

Shengchao Liu
Mila & Université de Montréal
liusheng@mila.quebec

Jieyu Zhang
University of Washington
jieyuz2@cs.washington.edu

Bolei Zhou
CUHK
bzhou@ie.cuhk.edu.hk

Abstract

Discovering new structures in the chemical space is a long-standing challenge and has important applications to various fields such as chemistry, material science, and drug discovery. Deep generative models have been used in *de novo* molecule design to embed molecules in a meaningful latent space and then sample new molecules from it. However, the steerability and interpretability of the learned latent space remain much less explored. In this paper, we introduce a new task named *molecule manipulation*, which aims to align the molecular properties of the generated molecule and its latent activation in order to achieve interactive molecule editing. Then we develop a method called **Chem**ical **Spac**e **E**xplorer (ChemSpacE), which identifies and traverses interpretable directions in the latent space that align with molecular structures and property changes. Specifically, ChemSpacE leverages the properties of the learned latent space by generative models and utilizes linear models to identify such directions and thus is highly efficient in terms of training/inference time, data, and the number of oracle calls. Experiments show that ChemSpacE can efficiently steer the latent spaces of multiple state-of-the-art molecule generative models for interactive molecule discovery.

1 Introduction

Designing new molecules with desired properties is a critical problem with a range of applications in drug discovery and material science [1]. Traditional pipelines require exhaustive human efforts and domain knowledge, which are difficult to scale up. Recent studies exploit deep generative models to solve this problem by encoding molecules into a meaningful latent space, from which random samples are drawn and decoded to new molecules [2]. Such deep molecule generative models facilitate the design and development of drugs and materials [3, 4].

Despite the promising results of deep generative models for molecule generation, much less effort has been made to understand the learned representations. Most of the existing models are based on deep neural networks, which are known to be black-box lacking transparency [5]. Outside of the molecule generation domain, many attempts have been made to improve the interpretability of deep learning models from various aspects, *e.g.*, representation space [6], model space [7], and latent space [8, 9]. In the molecule generation field, interpretability can be studied in two ways: (1) the interpretation of the **learned latent space** where traversing the value of latent vectors could lead to smooth molecular property change and (2) the interpretation of the **chemical space** that adjusting the molecular property could observe smooth structure change of molecules.

Preprint. Under review.

In addition, it remains challenging to generate molecules with desired properties. Previous works tackle the problem with latent space optimization-based, reinforcement learning-based, and searching-based methods to achieve property control of the generated molecules [10, 11]. Specifically, reinforcement learning-based algorithm [12] equips the model with rewards designed to encourage the molecule generative models to generate molecules with specific molecular properties. Latent space optimization-based algorithm takes advantage of the learnt latent space by molecule generative models and optimize the molecular properties via Bayesian Optimization [13]. Searching-based algorithm instead searches directly from the discrete and high-dimensional chemical space for molecules with optimal properties [14]. However, these work often have three major issues. (1) They require many times of expensive oracle calls to provide feedback (*i.e.*, property scores) of the intermediate molecules during the searching or optimization process [15]. (2) They often only focus on the outcome of the process while ignoring the intermediate steps of the process which is rather essential for chemists and pharmacologists to understand the chemical instances and rules. (3) They stick to local gradients while putting less focus on global directions in the chemical/latent space. (4) The molecular properties considered in the existing work are limited to a small set of molecular properties, such as penalized logP (octanol-water partition coefficient), QED (drug-likeness), DRD2 activity (binding affinity), *etc* [10, 11, 13, 16].

To tackle the above challenges, we formulate a new task, *molecule manipulation*, which aims at manipulating the properties of generated molecules by leveraging the steerability and interpretability of molecule generative models. Broad observations of the latent space learned by molecule generative models have been observed [17, 18]: (1) molecules sharing similar structures/properties tend to cluster in the latent space, (2) interpolating two molecules in the latent space lead to smooth changes in molecular structures/properties, we develop *ChemSpace Explorer*, a model-agnostic and efficient method to manipulate molecules with smooth changes of molecular structures and properties which has critical applications in molecule optimization, chemical space interpretation, etc. Specifically, *ChemSpace Explorer* first identifies the *property separation hyperplane* which defines the binary boundary corresponding to some molecular property (*e.g.*, drug-like or drug-unlike) in the learned latent space of a generative model. Based on the identified property separation hyperplane, we estimate the *latent directions* that govern molecular properties, which enable smooth change of the molecular structures and properties without re-training the given molecular generative model. To the best of our knowledge, this work is the first attempt to achieve interactive molecule discovery through the manipulating of pre-trained molecule generative models.

The experiments demonstrate that our method can effectively steer the state-of-the-art molecule generative models for molecule manipulation with a very small amount of training/inference time, data, and oracle calls. To quantitatively measure the performance of molecule manipulation, we design two new evaluation metrics named *strict success rate* and *relax success rate*, which evaluate the percentage of successful manipulations with smooth property-changing molecules over manipulations of a group of molecules. To facilitate the interactive molecule design and discovery for practitioners, we further develop an interface to visualize the real-time molecule manipulations and smooth molecular structure/property changes. Our main contributions are summarized as follows:

- We formulate *molecule manipulation*, a new task that steers the latent space of molecule generative models to manipulate the chemical properties of the output molecule.
- We develop an efficient model-agnostic method named *ChemSpace Explorer* for molecule manipulation, which can be incorporated in various state-of-the-art molecule generative models.
- Comprehensive experiments demonstrate the effectiveness of our method in quantifying the steerability of various molecule generative models. An interface is developed to exhibit the real-time molecule discovery and design.

2 Related Work

Molecule Generation. Recent studies have explored a variety of deep generative models for molecule generation [19], such as variational autoencoders (VAEs) [11], generative adversarial networks (GANs) [20], normalizing flows [10, 21, 22], energy-based models (EBMs) [23], reinforcement learning [24–26], *etc* [27, 28]. To be specific, JT-VAE [11] proposes a VAE-based architecture to encode both atomic graphs and structural graphs for efficient molecule generation. MolGAN [20] exploits GANs for molecule generation, where discriminators are used to encourage the model to

generate realistic and chemically-valid molecules. MRNN [29] extends the idea of GraphRNN [30] to formulate molecule generation as an auto-regressive process. GCPN [12] formulates the molecule generation process as a reinforcement learning problem where it obtains a molecule step by step by connecting atoms and reward is used for steerable generation. GraphNVP [21] first introduces normalizing flows for molecule generation, where the generation process is invertible. Later works improve the flow-based models via auto-regressive generation [10], valency correction [18], and discrete latent representation [22]. GraphEBM [23] introduces energy-based models based on the density of molecule data.

Controllable Molecule Generation. Another key point for molecule generation is to generate new molecular samples which possess certain properties. Early work [31] enforces bias on the distribution of the data and train the generative models with known desired properties to generate molecules with desired properties, while recent works mainly leverage latent space optimization-based [11, 12, 32, 33], reinforcement learning-based [10, 18, 34], and searching-based [14, 27, 35] approaches to generate molecules with desired properties. Latent space optimization-based methods are quite flexible and can work directly on both the molecules [] and the learned latent vectors [17, 33, 36–38]. Reinforcement learning-based methods usually formulate controllable generation as a sequential decision-making problem and require a score-function to reward the agent. Searching-based approaches [14, 16, 27, 28, 35, 39, 40] are also capable of searching over chemical space for molecules with desired properties by defining a set of discrete actions. Besides, a few works [41, 42] leverage the learned latent space and achieve controllable generation by accepting/rejecting sampled molecules based on a molecular property predictor. Despite the ability to generate molecules with desired properties, existing works usually suffer from the unrealistic assumption of the unlimited number of oracle calls [15]. Additionally, it is difficult to understand the generation process and the chemical rules that govern the generation process with previous methods [43].

3 Preliminaries

Molecule Graph. A molecule can be presented as a graph $X = (\mathcal{V}, \mathcal{E}, E, F)$, where V denotes a set of N vertices (*i.e.*, atoms), $\mathcal{E} \subseteq V \times V$ denotes a set of edges (*i.e.*, bonds), $F \in \{0,1\}^{N \times D}$ denotes the node features (*i.e.*, atom types) and $E \in \{0,1\}^{N \times N \times K}$ denotes the edge features (*i.e.*, bond types). The number of atom types and bond types are denoted by D and K, respectively.

Deep Molecule Generative Models. In molecule generation, a generative model M encodes the molecular graph X as a latent vector $Z \in \mathbb{R}^l$ with l being the dimension of the latent space and is capable of decoding any latent vector back to the molecular space. Specifically, variational auto-encoder (VAE) [44] and flow-based model (Flow) [45] are the two most commonly used models for molecule generation tasks. Both of them encode the data from molecular space to latent space, which is usually modeled as a Gaussian distribution; then they decode the latent code back to molecular space. They can be formulated as:

$$z = f(x), \qquad x' = g(z), \tag{1}$$

where x and x' are the ground-truth and reconstructed/sampled data respectively, and $z \in Z$ represents a latent vector in the latent space, $f(.)$ and $g(z)$ are the encoder and generator/decoder of the generative model.

4 Problem Formulation of Molecule Manipulation

To leverage the steerability and interpretability of molecule generative models, we propose a new task, *molecule manipulation*, which interprets and steer the latent space of the generative model in order to manipulate the properties of the output molecule. To be specific, a deep generative model contains a generator $g\colon \mathcal{Z} \to \mathcal{X}$, where $\mathcal{Z} \in \mathbf{R}^l$ stands for the l-dimensional latent space, which is commonly assumed to be Gaussian distribution [44, 45]. There exist property functions f_P which define the property space $\mathcal{P}$ via $P = f_P(X)$.

Formulation. The input to molecule manipulation is a list of n molecules $X = \{x_1, x_2, \cdots, x_n\}$ and a list of m molecular properties $P = \{p_1, p_2, \cdots, p_m\}$. We aim to manipulate one or more molecular properties p of a given molecule in a k consecutive steps and output the manipulated molecules with properties $p' = \{p^{(1)}, p^{(2)}, \cdots, p^{(k)}\}$. By manipulating the given molecule, we can observe the alignment of $\mathcal{Z} \to \mathcal{X} \to \mathcal{P}$, where the relationship between $\mathcal{Z}$ and $\mathcal{X}$ explains the latent

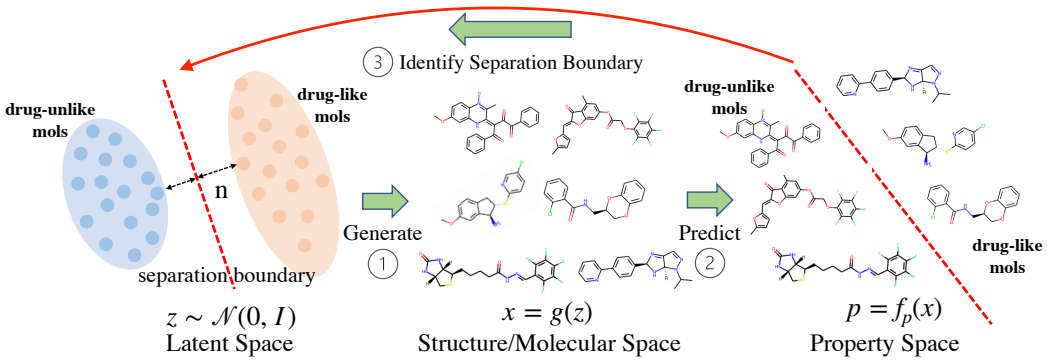


Figure 1: *ChemSpacE* framework: (1) the tested molecule generative model generates novel molecules by sampling random vector from the latent space and then feeding it into the generator, (2) off-the-shelf oracle function is used to predict molecular properties from the chemical space, (3) ChemSpacE identifies latent directions which govern molecular properties via the property separation hyperplane.

space of molecule generative models. The relationship between $\mathcal{X}$ and $\mathcal{P}$ reveals the correlations between molecular structures and properties. By traversing latent space, we can generate molecules with continuous structure/property changes.

Evaluation criteria. There are two important measures to evaluate the molecule manipulation task: smooth structure change and smooth property change. To be specific, we design two new evaluation metrics named *strict success rate (SSR)* and *relaxed success rate (RSR)* that measure the quality of the identified latent direction in controlling the molecular property. Under strict success rate, we consider a manipulation path to be successful only if we generate molecules with monotonically-changing properties and structures in consecutive k steps of manipulation. The constraints are formulated as follows:

$$\phi_{SPC}(x, k, f) = 1[\forall\, i \in [k], s.t., f(x^{(i)}) - f(x^{(i+1)}) \leq 0], \tag{2}$$

$$\phi_{SSC}(x, k, \delta) = 1[\forall\, i \in [k], s.t., \delta(x^{(i+1)}, x^{(1)}) - \delta(x^{(i)}, x^{(1)}) \leq 0], \tag{3}$$

$$\phi_{DIV}(x, k) = 1[\exists\, i \in [k], s.t., x^{(i)} \neq x^{(1)}], \tag{4}$$

where f is a property function which calculates certain molecular property, δ denotes structure similarity between molecules $x^{(i)}$, $x^{(i+1)}$ generated in two adjacent manipulation steps. ϕ_{SPC} defines the strict property constraint; ϕ_{SSC} defines the strict structure constraint; ϕDIV defines the diversity constraint. The strict success rate is defined as:

$$SSR - L(P, X, k) = \frac{1}{|P| \times |X|} \sum\nolimits_{p \in P, x \in X} 1[\phi_{SPC}(x_p, k, f_p) \wedge \phi_{SSC}(x_p, k) \wedge \phi_{DIV}(x_p, k)], \tag{5}$$

As monotonicity is rather strict, we propose a more relaxed definition of success rate, namely relaxed success rate, constructed via relaxed constraints, as follows:

$$\phi_{RPC}(x, k, f, \epsilon) = 1[\forall\, i \in [k], s.t., f(x^{(i)}) - f(x^{(i+1)}) \leq \epsilon], \tag{6}$$

$$\phi_{RSC}(x, k, \delta, \gamma) = 1[\forall\, i \in [k], s.t., \delta(x^{(i+1)}, x^{(1)}) - \delta(x^{(i)}, x^{(1)}) \leq \gamma], \tag{7}$$

$$\phi_{DIV}(x, k) = 1[\exists\, i \in [k], s.t., x^{(i)} \neq x^{(1)}], \tag{8}$$

where ϵ is a predefined tolerance threshold that weakens the monotonicity requirement. We also provide two implementations of relaxed success rate, which defines different tolerance variables ϵ with local relaxed constraint (RSR-L) and global relaxed constraint (RSR-G). For global constraint, we obtain ϵ by calculating the possible values (ranges) of the molecular properties in the training dataset, while for local constraint, we obtain ϵ by calculating the possible values (ranges) of the molecular properties only in the specific manipulation paths. The formulation of RSR-L and RSR-G is as follows:

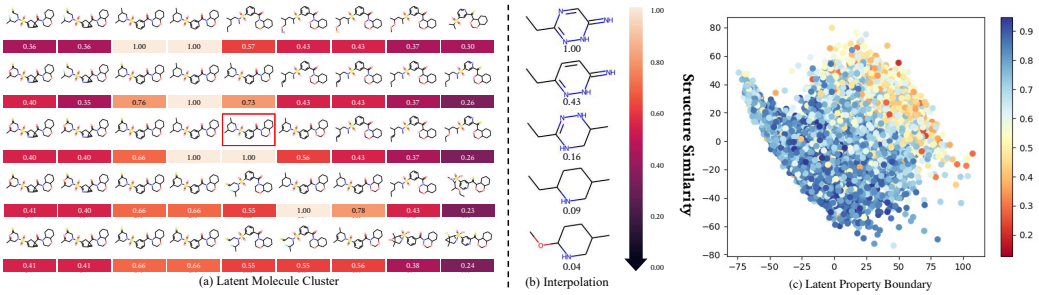


Figure 2: (a) Molecule clusters in the latent space, the number represents structure similarity [46], where the red box represents the base molecule, x and y axes denote two random orthogonal directions to manipulate. (b) Linear interpolation of two (top and bottom) molecules. (c) Latent property boundary is visualized for QED property.

$$RSR-L(P,X,k,\epsilon_l,\gamma) = \frac{1}{|P|\times|X|}\sum\nolimits_{p\in P,x\in X} 1[\phi_{RPC}(x_p,k,f_p,\epsilon_l)\wedge\phi_{RSC}(x_p,k,\gamma)\wedge\phi_{DIV}(x_p,k)], \tag{9}$$

$$RSR-G(P,X,k,\epsilon_g,\gamma) = \frac{1}{|P|\times|X|}\sum\nolimits_{p\in P,x\in X} 1[\phi_{RPC}(x_p,k,f_p,\epsilon_g)\wedge\phi_{RSC}(x_p,k,\gamma)\wedge\phi_{DIV}(x_p,k)], \tag{10}$$

Note even though it is more challenging for the model to pass RSR-L with local constraint (smaller range) while evaluating the successful path, its extra benefit is to take into account the ability of the model to manipulate one molecular property (*i.e.*, the larger the range, the higher the tolerance score, thus the better chance to achieve successful manipulation).

5 ChemSpacE for Molecule Manipulation

5.1 Latent Cluster Assumption

We examine the property of latent space learned by the generative models and have the following observations, (1) molecules with similar structures tend to cluster in the latent space, (2) interpolating two molecules x_1 and x_2, represented by latent vectors z_1 and z_2, can lead to a list of intermediate molecules whose structures/properties gradually change from x_1 to x_2. As molecular structures determine molecular properties [47], the observations imply that molecules with similar property values of certain molecular property would cluster together and interpolating two molecules with different values of the molecular property could lead to gradual changes in molecular structures. As shown in Fig. 1, there may exist two groups of molecules, drug-like and drug-unlike, where each group cluster together and linear interpolating two latent vectors with one molecule from each group could lead to a direction that crosses the property separation boundary. These observations also match the analysis from the prior work [17, 18]. To verify our assumption, we visualize the latent space of the pre-trained MoFlow model in Fig. 2. The left figure shows that molecules close in the latent space are similar in structures. The middle figure shows that interpolating two molecules in the latent space could lead to smooth structure changes. The right figure shows that the latent boundary is present for QED property in the pre-trained MoFlow model.

5.2 Identifying Latent Directions

Latent Separation Boundary. With the verifications above and the previous work of analyzing the latent space of generative models [8, 48–50], we assume that there exists a separation boundary which separates groups of molecules for each molecular property (*e.g.*, drug-like and drug-unlike) and the normal vector of the separation boundary defines a latent direction which controls the degree of the property value (in Fig. 1). When z moves toward and crosses the boundary, the molecular properties change accordingly (*e.g.*, from drug-unlike to drug-like). A perfect separation boundary would have molecules with different properties well separated on different sides. From that, we can find a separation boundary for each molecular property with a unit normal vector $n \in \mathbf{R}^l$, such that

Table 1: Quantitative Evaluation of Molecule Manipulation over a variety of molecular properties (numbers reported are *strict success rate* in %, -R denotes model with random manipulation, -L denotes model with the largest range manipulation, -O denotes optimization-based manipulation, -C denotes model with ChemSpacE. The best performances are bold.

Datasets	Models	Avg.	QED	LogP	SA	DRD2	JNK3	GSK3B	MolWt
QM9	MoFlow-R	1.65	1.50	0.00	0.50	0.00	0.00	0.00	0.50
	MoFlow-L	3.43	1.50	1.00	0.50	0.00	1.50	0.00	0.50
	MoFlow-O	N/A	3.50	6.00	6.50	2.00	8.00	8.50	7.50
	MoFlow-C	**37.52**	**12.50**	**9.00**	**10.00**	**11.00**	**45.50**	**16.50**	**10.50**
	HierVAE-R	29.29	1.00	1.50	0.50	0.50	1.00	1.00	0.50
	HierVAE-L	30.69	0.50	0.00	0.00	0.50	2.00	0.00	0.50
	HierVAE-C	**66.23**	**27.00**	**32.00**	**35.00**	**41.50**	**51.50**	**30.00**	**39.50**
ZINC	MoFlow-R	4.25	1.50	1.50	2.50	3.00	3.50	1.50	2.00
	MoFlow-L	5.61	1.50	6.50	2.00	6.00	2.50	4.00	1.50
	MoFlow-O	N/A	1.50	9.50	0.50	2.00	15.50	23.00	0.00
	MoFlow-C	**58.08**	**52.00**	**53.50**	**51.50**	**55.00**	**56.50**	**55.50**	**53.50**
ChEMBL	HierVAE-R	25.59	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	HierVAE-L	22.98	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	HierVAE-C	**47.70**	**0.50**	**3.00**	**3.00**	**6.00**	**7.50**	**5.50**	**4.50**

the distance from any sample z to the separation boundary as:

$$d(z, n) = n^T z \tag{11}$$

Latent Direction. In the latent space, the molecular structure and property change smoothly towards the new property class when z moves towards the separation boundary and vice versa, where we assume linear dependency between z and p:

$$f_P(g(z)) = \alpha \cdot d(z, n), \tag{12}$$

where f_P is an oracle function and α is a degree scalar that scales the changes along that corresponding direction. Extending the method to multiple molecular properties manipulation, we have:

$$f_P(g(z)) = AN^T z, \tag{13}$$

where $A = Diag(a_1, \cdots, a_m)$ is the diagonal matrix with linear coefficients for each of the m molecular properties and $N = [n_1, \cdots, n_m]$ represents normal vectors for the separation boundaries of m molecular properties. We have the molecular properties P following a multivariate normal distribution via:

$$\mu_P = \mathbf{E}(AN^T z) = AN^T \mathbf{E}(z) = \mathbf{0}, \tag{14}$$

$$\Sigma_P = \mathbf{E}(AN^T z z^T N A^T) = AN^T \mathbf{E}(zz^T) N A^T = AN^T N A^T. \tag{15}$$

We have all disentangled molecular properties in P if and only if Σ_P is a diagonal matrix and all directions in N are orthogonal with each other. Nevertheless, not all molecular properties are purely disentangled with each other. In that case, molecular properties can correlate with each other and $n_i^T n_j$ is used to denote the entanglement between the i-th and j-th molecular properties in P.

5.3 Molecule Manipulation

After we find the separation boundary and identify the latent direction, to manipulate the generated molecules with desired properties, we first move from latent vector z along the direction n with a degree scalar α, and the new latent vector is

$$z' = z + \alpha n \tag{16}$$

To this end, the expected property of the new manipulated molecule is

$$f_P(g(z + \alpha n)) = f_P(g(z)) + k\alpha, \tag{17}$$

where k is a scaling factor between molecular vector space and property. Based on our assumption to find a separation boundary for each molecular property, we could utilize any linear model (*e.g.* linear Support Vector Machine) [51] to find the separation boundaries which best separate the two classes of the data. For each molecular property, we train an individual model from a group of randomly sampled latent vectors and utilize a property function f_P to calculate the corresponding molecular properties. Then, we find the separation boundary for each molecular property. The normal vectors N of separation boundaries are finally utilized as identified latent directions that govern the molecular properties. Additionally, our method is highly efficient in terms of data, training time and offline oracle calls thanks to leveraging shallow models with only a small number of data and their pre-calculated molecular properties.

6 Experiments

6.1 Setup

Datasets. We use three molecule datasets, QM9 [52], ZINC250K [53], and ChEMBL [54]. QM9 contains 134k small organic molecules with up to 9 heavy atoms (C, O, N, F). ZINC is a free database of commercially-available compounds for drug discovery. On average, the molecules in ZINC are bigger ($\sim$23 heavy atoms) and structurally more complex than QM9. We take a sampled 250K molecules version [17] from the larger database. ChEMBL is a manually curated database of bioactive molecules with drug-like properties and contains $\sim$1.8 million molecules.

Baselines. We include two baseline methods of identifying latent direction that governs the molecular property and one optimization-based method, which optimizes the molecular property of the generated molecules via gradient ascent/descent for comparisons. **Random manipulation** randomly samples latent directions for molecular properties. **Largest range manipulation** draws latent vectors from the training set and defines the directions via calculating the direction between one molecule with the largest property score and another molecule with the smallest property score for each molecular property. **Optimization-based method** optimizes the molecular property of the generated molecules by searching a latent vector with the optimized molecular property via gradient ascent/descent.

Implementation Details. We take the publicly available pre-trained models from the GitHub Repository of HierVAE and MoFlow, respectively. We utilize the implementation of linear models (linear SVM) from Scikit-learn [55].

6.2 Evaluation Protocols

Pre-trained Models. We apply ChemSpacE, as well as baselines, on two state-of-the-art molecule generative models with publicly available pre-trained models. HierVAE [56] embeds molecular structure motifs into a hierarchical VAE-based generative model; MoFlow [18] designs a normalizing flow-based model which learns an invertible mapping between input molecules and latent vectors.
Molecular Properties. We study molecular properties identified in the chemistry community through open-source cheminformatics software, RDKit [57] and protein binding affinity, synthesis accessibility oracles in TDC [15]. In total, we analyze 212 molecular properties from multiple dimensions, including distributions, inter-correlations, etc. Details can be found in Appendix B. Due to the page limit, we mainly report results for 7 molecular properties, including 4 very common yet important ones, drug-likeness (QED), molecular weight (MolWt), partition coefficient (LogP), synthesis accessibility (SA), and 3 binding affinity scores. For continuous molecular properties, we take the molecules with largest and smallest properties for training the linear models.

Quantitatively, we evaluate the ability of the model to manipulate the given molecular property of molecules with the proposed **strict success rate** and **relaxed success rate-L/G** metrics (see Sec. 4). We evaluate the model's efficiency by comparing the training process of the linear models with a neural network-based predictor for a commonly used optimization-based method in terms of training/inference time, data, and number of oracle calls. Qualitatively, we visualize molecule manipulation including property distribution shift during manipulation, single and multiple property manipulations.

Table 2: Efficiency in terms of training/inference time, data, and number of oracles of ChemSpacE compared to the optimization-based method.

Model	Dataset	Training(s)	Inference/Path(s)	# Data	# Oracle calls
Opt-based	QM9	137.03	0.02	120k	120k
	ZINC	1027.26	0.04	200k	200k
ChemSpacE	QM9	0.05	0	300	300
	ZINC	0.95	0	400	400
Speedup	QM9	2740×	0.02 ↑	400×	400×
	ZINC	1080×	0.04 ↑	500×	500×

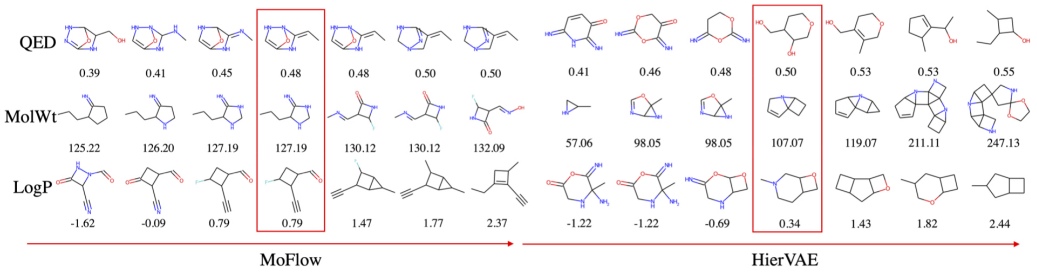


Figure 3: Manipulating QED, MolWt and LogP properties of sampled molecules. The backbone model is CGVAE trained on QM9 dataset.

6.3 Quantitative Evaluation of Molecule Manipulation

In Table 1 and 2, we report the quantitative evaluation results for molecule manipulation with both strict success rate and relaxed success rate-L/G and training, inference time, data, oracle calls efficiency, which are evaluated on 212 molecular properties over $1,000$ randomly generated molecules. According to the table, we can obtain the following insights.

(1) Our proposed method, ChemSpacE, as the first attempt for molecule manipulation, achieves excellent performance to manipulate properties of molecules with two state-of-the-art molecule generative models (VAE-based and Flow-based). For some important molecular properties (*e.g.*, QED), we (with MoFlow) achieve 52% manipulation strict success rate in ZINC dataset. We outperform the baseline methods $6\times$ on average.

(2) The baseline (random manipulation) method sometimes "finds" directions that control molecular properties. As shown in Fig. 2, the molecules are well-clustered in the latent space with respect to structures that determine molecular properties [47]. However, the largest range manipulation works worse possibly due to its strong assumption in determining the direction via the molecules with extreme properties (largest property and smallest property) in the dataset.

(3) The ChemSpacE method outperforms the popular optimization-based method in both generating smooth manipulation path, time and data efficiency. In Table 2, ChemSpacE speeds up the training time for at least $1000\times$, required data for at least $400\times$, and required oracle calls for at least $400\times$.

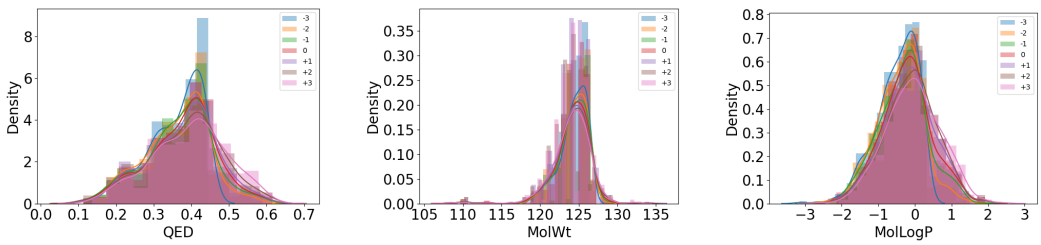


Figure 4: Visualization of Molecular property distribution shift while manipulating molecules with MoFlow on QM9 dataset (0 denotes the randomly sampled base molecule and $+x$ and $-x$ denote manipulation directions and steps).

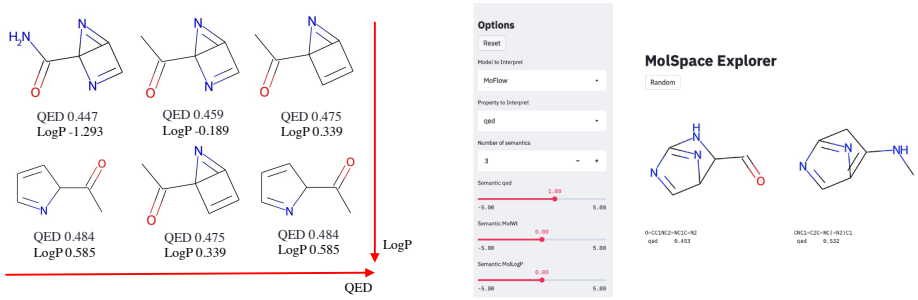


Figure 5: Manipulating QED and LogP properties of sampled molecules simultaneously with MoFlow model trained on QM9 dataset (the repeated molecules are removed for better visualization) (left). A Real-time Interactive System Interface. Please refer to Appendix E demo video for interactive molecule discovery (right).

6.4 Qualitative Evaluation of Molecule Manipulation and Interpretation

In Fig. 4, we visualize the property distributions of QED, MolWt and LogP along a 7-step manipulation path. For each step, we draw a property distribution. The candidate molecules are at place 0 and we attempt to manipulate the molecular property to the left (lower) and the right (higher). From the figure, we can clearly observe that the property distribution shifts to the left and right accordingly when we manipulate the molecule to the left and right. For example, when we manipulate the molecules three steps to the left, the range of QED shifts from $[0, 0.7]$ to $[0, 0.5]$; when the molecules are manipulated three steps to the right, there are much more molecules that have QED > 0.5 than the base distribution. Similar trends can also be seen for MolWt and LogP properties.

Single Property Manipulation. To qualitatively evaluate the performance of our method for molecule manipulation, we randomly select the successful manipulation paths from all three generative models in Fig. 3. The figures show that our method successfully learns interpretable and steerable directions. For example, for HierVAE in Fig. 3, we can find that gradually increasing LogP of a molecule may lead to the removal of the heavy atoms O and N from the structure. With respect to QED, the molecule drops double bonds, as well as heavy N and O atoms, when increasing QED for the HierVAE model. A similar trend can be observed in the MoFlow model that increasing QED drops double bonds and O atoms on the left of Fig. 3.

Multi-Property Manipulation. When it comes to multi-property manipulation, the goal is to control multiple molecular properties of a given molecule at the same time. In Fig. 5 (left), we show how our method manipulates multiple molecular properties. For simplicity, we remove the duplicate molecules and only leave the distinct molecules during the manipulation. From the figure, we can observe some correlations between LogP and QED since when we increase QED, LogP also increases accordingly. However, it is not always the case as moving the molecules to the right in the second row does not increase the QED scores. One potential reason is that the chemical space is vast, discrete and complex, and it is nontrivial to manipulate only one property while keep others the same of a molecule. An interactive demo is provided at `https://drive.google.com/drive/folders/1N036p_5OfvGZybgPJ3Vw1ONXHVepimSR?usp=sharing` and shown in Fig. 5 (right).

7 Conclusion

In this work, we formulate a new task of molecule manipulation and develop an efficient method called ChemSpacE to improve the steerability and interpretability of molecular generative models. The interface illustrates the promising application of interactive molecule design and discovery. Nevertheless, some challenging chemical phenomenons, such as activity cliff, are not yet studied in this work. In addition, exploring the chemical space with unbiased pre-trained generative models is still nascent, it is also worth discovering a biased latent space for more effective molecule manipulation in our future work.

References

[1] Hongming Chen, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke. The rise of deep learning in drug discovery. *Drug discovery today*, 23(6):1241–1250, 2018. 1

[2] W Patrick Walters and Regina Barzilay. Applications of deep learning in molecule generation and molecular property prediction. *Accounts of Chemical Research*, 54(2):263–270, 2020. 1

[3] Romain Lopez, Adam Gayoso, and Nir Yosef. Enhancing scientific discoveries in molecular biology with deep generative models. *Molecular Systems Biology*, 16(9):e9198, 2020. 1

[4] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, 2018. 1

[5] Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature, 2019. 1

[6] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 1

[7] Xiaojie Guo, Yuanqi Du, and Liang Zhao. Deep generative models for spatial networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 505–515, 2021. 1

[8] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1, 5

[9] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1532–1540, 2021. 1

[10] Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382*, 2020. 2, 3, 14

[11] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, pages 2323–2332. PMLR, 2018. 2, 3, 14

[12] Jiaxuan You, Bowen Liu, Rex Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *arXiv preprint arXiv:1806.02473*, 2018. 2, 3, 14

[13] Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander L Gaunt. Constrained graph variational autoencoders for molecule design. *arXiv preprint arXiv:1805.09076*, 2018. 2, 14

[14] Youngchun Kwon, Seokho Kang, Youn-Suk Choi, and Inkoo Kim. Evolutionary design of molecules based on deep learning and a genetic algorithm. *Scientific reports*, 11(1):1–11, 2021. 2, 3

[15] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *Advances in neural information processing systems*, 2021. 2, 3, 7

[16] Tianfan Fu, Cao Xiao, and Jimeng Sun. Core: Automatic molecule optimization using copy & refine strategy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 638–645, 2020. 2, 3

[17] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018. 2, 3, 5, 7

[18] Chengxi Zang and Fei Wang. Moflow: an invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 617–626, 2020. 2, 3, 5, 7, 14

[19] Yuanqi Du, Tianfan Fu, Jimeng Sun, and Shengchao Liu. Molgensurvey: A systematic survey in machine learning models for molecule design. *arXiv preprint arXiv:2203.14500*, 2022. 2

[20] Nicola De Cao and Thomas Kipf. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018. 2

[21] Kaushalya Madhawa, Katushiko Ishiguro, Kosuke Nakago, and Motoki Abe. Graphnvp: An invertible flow model for generating molecular graphs. *arXiv preprint arXiv:1905.11600*, 2019. 2, 3, 14

[22] Youzhi Luo, Keqiang Yan, and Shuiwang Ji. Graphdf: A discrete flow model for molecular graph generation. *arXiv preprint arXiv:2102.01189*, 2021. 2, 3, 14

[23] Meng Liu, Keqiang Yan, Bora Oztekin, and Shuiwang Ji. Graphebm: Molecular graph generation with energy-based models. *arXiv preprint arXiv:2102.00546*, 2021. 2, 3, 14

[24] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):1–14, 2017. 2

[25] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):1–10, 2019.

[26] Soojung Yang, Doyeong Hwang, Seul Lee, Seongok Ryu, and Sung Ju Hwang. Hit and lead discovery with explorative rl and fragment-based molecule generation. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[27] Xiufeng Yang, Tanuj Kr Aasawat, and Kazuki Yoshizoe. Practical massively parallel monte-carlo tree search applied to molecular design. *arXiv preprint arXiv:2006.10504*, 2020. 2, 3

[28] Yutong Xie, Chence Shi, Hao Zhou, Yuwei Yang, Weinan Zhang, Yong Yu, and Lei Li. Mars: Markov molecular sampling for multi-objective drug discovery. *arXiv preprint arXiv:2103.10432*, 2021. 2, 3

[29] Mariya Popova, Mykhailo Shvets, Junier Oliva, and Olexandr Isayev. Molecularrnn: Generating realistic molecular graphs with optimized properties. *arXiv preprint arXiv:1905.13372*, 2019. 3, 14

[30] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International conference on machine learning*, pages 5708–5717. PMLR, 2018. 3

[31] Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1): 120–131, 2018. 3

[32] Samuel Hoffman, Vijil Chenthamarakshan, Kahini Wadhawan, Pin-Yu Chen, and Payel Das. Optimizing molecules using efficient queries from property evaluations. *arXiv preprint arXiv:2011.01921*, 2020. 3

[33] Robin Winter, Floriane Montanari, Andreas Steffen, Hans Briem, Frank Noé, and Djork-Arné Clevert. Efficient multi-objective molecular optimization in a continuous latent space. *Chemical science*, 10(34):8016–8024, 2019. 3

[34] Thomas Blaschke, Josep Arús-Pous, Hongming Chen, Christian Margreitter, Christian Tyrchan, Ola Engkvist, Kostas Papadopoulos, and Atanas Patronov. Reinvent 2.0: an ai tool for de novo drug design. *Journal of Chemical Information and Modeling*, 60(12):5918–5922, 2020. 3

[35] Nathan Brown, Marco Fiscato, Marwin HS Segler, and Alain C Vaucher. Guacamol: benchmarking models for de novo molecular design. *Journal of chemical information and modeling*, 59(3):1096–1108, 2019. 3

[36] Wengong Jin, Kevin Yang, Regina Barzilay, and Tommi Jaakkola. Learning multimodal graph-to-graph translation for molecular optimization. *arXiv preprint arXiv:1812.01070*, 2018. 3

[37] Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. Constrained bayesian optimization for automatic chemical design using variational autoencoders. *Chemical science*, 11(2):577–586, 2020.

[38] Pascal Notin, José Miguel Hernández-Lobato, and Yarin Gal. Improving black-box optimization in vae latent space using decoder uncertainty. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 3

[39] Philipp Renz, Dries Van Rompaey, Jörg Kurt Wegner, Sepp Hochreiter, and Günter Klambauer. On failure modes in molecule generation and optimization. *Drug Discovery Today: Technologies*, 32:55–63, 2019. 3

[40] Krzysztof Maziarz, Henry Jackson-Flux, Pashmina Cameron, Finton Sirockin, Nadine Schneider, Nikolaus Stiefl, Marwin Segler, and Marc Brockschmidt. Learning to extend molecular scaffolds with structural motifs. *arXiv preprint arXiv:2103.03864*, 2021. 3

[41] Vijil Chenthamarakshan, Payel Das, Samuel C Hoffman, Hendrik Strobelt, Inkit Padhi, Kar Wai Lim, Benjamin Hoover, Matteo Manica, Jannis Born, Teodoro Laino, et al. Cogmol: target-specific and selective drug design for covid-19 using deep generative models. *arXiv preprint arXiv:2004.01215*, 2020. 3

[42] Payel Das, Tom Sercu, Kahini Wadhawan, Inkit Padhi, Sebastian Gehrmann, Flaviu Cipcigan, Vijil Chenthamarakshan, Hendrik Strobelt, Cicero Dos Santos, Pin-Yu Chen, et al. Accelerated antimicrobial discovery via deep generative models and molecular dynamics simulations. *Nature Biomedical Engineering*, 5(6):613–623, 2021. 3

[43] Yuanqi Du, Xiaojie Guo, Amarda Shehu, and Liang Zhao. Interpretable molecule generation via disentanglement learning. In *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 1–8, 2020. 3

[44] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[45] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015. 3

[46] Dávid Bajusz, Anita Rácz, and Károly Héberger. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *J. Cheminformatics*, 7:20:1–20:13, 2015. doi: 10.1186/s13321-015-0069-3. URL https://doi.org/10.1186/s13321-015-0069-3. 5

[47] Paul G. Seybold, Michael May, and Ujjvala A. Bagal. Molecular structure: Property relationships. *Journal of Chemical Education*, 64(7):575, 1987. doi: 10.1021/ed064p575. URL https://doi.org/10.1021/ed064p575. 5, 8

[48] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017. 5

[49] Ali Jahanian, Lucy Chai, and Phillip Isola. On the" steerability" of generative adversarial networks. *arXiv preprint arXiv:1907.07171*, 2019.

[50] Antoine Plumerault, Hervé Le Borgne, and Céline Hudelot. Controlling generative models with continuous factors of variations. *arXiv preprint arXiv:2001.10238*, 2020. 5

[51] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995. 7

[52] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014. 7

[53] John J Irwin and Brian K Shoichet. Zinc- a free database of commercially available compounds for virtual screening. *Journal of chemical information and modeling*, 45(1):177–182, 2005. 7

[54] David Mendez, Anna Gaulton, A Patrícia Bento, Jon Chambers, Marleen De Veij, Eloy Félix, María Paula Magariños, Juan F Mosquera, Prudence Mutowo, Michał Nowotka, et al. Chembl: towards direct deposition of bioassay data. *Nucleic acids research*, 47(D1):D930–D940, 2019. 7

[55] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011. 7

[56] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Hierarchical generation of molecular graphs using structural motifs. In *International Conference on Machine Learning*, pages 4839–4848. PMLR, 2020. 7, 14

[57] Greg Landrum et al. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling, 2013. 7

[58] Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018. 16

[59] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *International Conference on Machine Learning*, pages 1945–1954. PMLR, 2017. 16

Appendix for "ChemSpacE: Toward Steerable and Interpretable Chemical Space Exploration"

A Molecule Generative Models

In Table 3, we summarize a list of representative molecule generative models, which span various types of deep generative models, including the type of generative models, the type of generation process and whether latent space is learned. We also provide the formulation for two types of deep generative models (VAE and Flow) in Section A that are very popular for molecule generation task.

Table 3: A summary of mainstream molecule generative models.

Prior Work	Generative Model	Sequential	Latent Space
JT-VAE [11]	VAE	✓	✓
CGVAE [13]	VAE	✓	✓
MRNN [29]	RNN	✓	
GraphNVP [21]	Flow		✓
GCPN [12]	RL	✓	
GraphAF [10]	Flow	✓	
MoFlow [18]	Flow		✓
HierVAE [56]	VAE	✓	✓
GraphEBM [23]	EBM		
GraphDF [22]	Flow	✓	

A.1 Molecule Generative Model Formulation

VAE. gets a lower bound (ELBO) for the data log probability by introducing a proposal distribution.

$$\begin{aligned} \log p(x) &= \log \int_z p(x|z)p(z)dz \\ &\geq \log[\mathbb{E}_{q(z|x)}[p(x|z)] + \mathrm{KL}(q(z|x)||p(z))] \end{aligned} \tag{18}$$

Flow. The key of Flow model is to design a invertible function with the following nice property:

$$\begin{aligned} &z_0 \sim p_0(z_0) \\ &z_i = f_i(z_{i-1}) \\ &z_{i-1} = f_i^{-1}(z_i) \\ &p_i(z_i) = p_{i-1}(z_{i-1})\Big|\det \frac{df_i^{-1}}{dz_i}\Big| = p_{i-1}(f_i^{-1}(z_i))\Big|\det \frac{df_i^{-1}}{dz_i}\Big|, \end{aligned} \tag{19}$$

where f_i is invertible function. To be more concrete, we can take z_0 as some tractable noise distribution, like Gaussian distribution, and repeating this for K steps will lead to the data distribution, *i.e.*,:

$$x = z_K = f_K \circ f_{K-1} \circ ... \circ f_1(z_0).$$

Thus, the log likelihood of the data is as follows:

$$\begin{aligned} \log p(x) &= \log p_K(z_K) \\ &= \log p_{K-1}(z_{K-1}) - \log \Big|\det \frac{df_K}{dz_{K-1}}\Big| \\ &= \log p_{K-2}(z_{K-2}) - \log \Big|\det \frac{df_{K-1}}{dz_{K-2}}\Big| - \log \Big|\det \frac{df_K}{dz_{K-1}}\Big| \\ &= ... \\ &= \log p_0(z_0) - \sum_{i=1}^{K} \log \Big|\det \frac{df_i}{dz_{i-1}}\Big| \end{aligned} \tag{20}$$

B Study of Molecular Properties

List of Molecular Properties. In total, study 208 molecular properties from the open chemistry library RDKit[1] and 4 important molecular properties including synthesis accessibility and binding affinity scores from TDC[2]. They are MaxEStateIndex, MinEStateIndex, MaxAbsEStateIndex, MinAbsEStateIndex, qed, MolWt, HeavyAtomMolWt, ExactMolWt, NumValenceElectrons, NumRadicalElectrons, MaxPartialCharge, MinPartialCharge, MaxAbsPartialCharge, MinAbsPartialCharge, FpDensityMorgan1, FpDensityMorgan2, FpDensityMorgan3, BCUT2D_MWHI, BCUT2D_MWLOW, BCUT2D_CHGHI, BCUT2D_CHGLO, BCUT2D_LOGPHI, BCUT2D_LOGPLOW, BCUT2D_MRHI, BCUT2D_MRLOW, BalabanJ, BertzCT, Chi0, Chi0n, Chi0v, Chi1, Chi1n, Chi1v, Chi2n, Chi2v, Chi3n, Chi3v, Chi4n, Chi4v, HallKierAlpha, Ipc, Kappa1, Kappa2, Kappa3, LabuteASA, PEOE_VSA1, PEOE_VSA10, PEOE_VSA11, PEOE_VSA12, PEOE_VSA13, PEOE_VSA14, PEOE_VSA2, PEOE_VSA3, PEOE_VSA4, PEOE_VSA5, PEOE_VSA6, PEOE_VSA7, PEOE_VSA8, PEOE_VSA9, SMR_VSA1, SMR_VSA10, SMR_VSA2, SMR_VSA3, SMR_VSA4, SMR_VSA5, SMR_VSA6, SMR_VSA7, SMR_VSA8, SMR_VSA9, SlogP_VSA1, SlogP_VSA10, SlogP_VSA11, SlogP_VSA12, SlogP_VSA2, SlogP_VSA3, SlogP_VSA4, SlogP_VSA5, SlogP_VSA6, SlogP_VSA7, SlogP_VSA8, SlogP_VSA9, TPSA, EState_VSA1, EState_VSA10, EState_VSA11, EState_VSA2, EState_VSA3, EState_VSA4, EState_VSA5, EState_VSA6, EState_VSA7, EState_VSA8, EState_VSA9, VSA_EState1, VSA_EState10, VSA_EState2, VSA_EState3, VSA_EState4, VSA_EState5, VSA_EState6, VSA_EState7, VSA_EState8, VSA_EState9, FractionCSP3, HeavyAtomCount, NHOHCount, NOCount, NumAliphaticCarbocycles, NumAliphaticHeterocycles, NumAliphaticRings, NumAromaticCarbocycles, NumAromaticHeterocycles, NumAromaticRings, NumHAcceptors, NumHDonors, NumHeteroatoms, NumRotatableBonds, NumSaturatedCarbocycles, NumSaturatedHeterocycles, NumSaturatedRings, RingCount, MolLogP, MolMR, fr_Al_COO, fr_Al_OH, fr_Al_OH_noTert, fr_ArN, fr_Ar_COO, fr_Ar_N, fr_Ar_NH, fr_Ar_OH, fr_COO, fr_COO2, fr_C_O, fr_C_O_noCOO, fr_C_S, fr_HOCCN, fr_Imine, fr_NH0, fr_NH1, fr_NH2, fr_N_O, fr_Ndealkylation1, fr_Ndealkylation2, fr_Nhpyrrole, fr_SH, fr_aldehyde, fr_alkyl_carbamate, fr_alkyl_halide, fr_allylic_oxid, fr_amide, fr_amidine, fr_aniline, fr_aryl_methyl, fr_azide, fr_azo, fr_barbitur, fr_benzene, fr_benzodiazepine, fr_bicyclic, fr_diazo, fr_dihydropyridine, fr_epoxide, fr_ester, fr_ether, fr_furan, fr_guanido, fr_halogen, fr_hdrzine, fr_hdrzone, fr_imidazole, fr_imide, fr_isocyan, fr_isothiocyan, fr_ketone, fr_ketone_Topliss, fr_lactam, fr_lactone, fr_methoxy, fr_morpholine, fr_nitrile, fr_nitro, fr_nitro_arom, fr_nitro_arom_nonortho, fr_nitroso, fr_oxazole, fr_oxime, fr_para_hydroxylation, fr_phenol, fr_phenol_noOrthoHbond, fr_phos_acid, fr_phos_ester, fr_piperdine, fr_piperzine, fr_priamide, fr_prisulfonamd, fr_pyridine, fr_quatN, fr_sulfide, fr_sulfonamd, fr_sulfone, fr_term_acetylene, fr_tetrazole, fr_thiazole, fr_thiocyan, fr_thiophene, fr_unbrch_alkane, fr_urea, sa, drd2, jnk3, gsk3b.

However, not all of the molecular properties are varied in the three datasets. Specifically, **QM9** contains 29 frozen molecular properties, NumRadicalElectrons, SMR_VSA8, SlogP_VSA12, SlogP_VSA7, SlogP_VSA9, EState_VSA11, VSA_EState10, fr_C_S, fr_N_O, fr_SH, fr_azide, fr_azo, fr_barbitur, fr_benzodiazepine, fr_diazo, fr_hdrzine, fr_hdrzone, fr_isocyan, fr_isothiocyan, fr_nitroso, fr_phos_acid, fr_phos_ester, fr_prisulfonamd, fr_sulfide, fr_sulfonamd, fr_sulfone, fr_thiazole, fr_thiocyan, fr_thiophene, **ZINC** contains 4 frozen molecular properties, NumRadicalElectrons, SMR_VSA8, SlogP_VSA9, fr_prisulfonamd and **ChEMBL** contains only 3 frozen molecular properties, SMR_VSA8, SlogP_VSA9, fr_prisulfonamd.

Inter-correlations of molecular properties. In Fig. 6, we visualize the linear correlations between each pair of molecular properties across three datasets. From the heatmaps, we can observe that there are no linear correlations between half of the molecular properties, and similar patterns are observed in ZINC and ChEMBL datasets.

Molecular Property Distributions. We visualize 7 molecular property distributions reported in section 6 in Fig. 7. From there, we can observe that the property distribution may vary a lot in terms of different datasets. Notably, the distributions of some properties, *e.g.*, QED, are very similar in ZINC and ChEMBL datasets, while some are quite different, *e.g.*, MolWt.

[1] https://www.rdkit.org/docs/index.html

[2] https://tdcommons.ai/

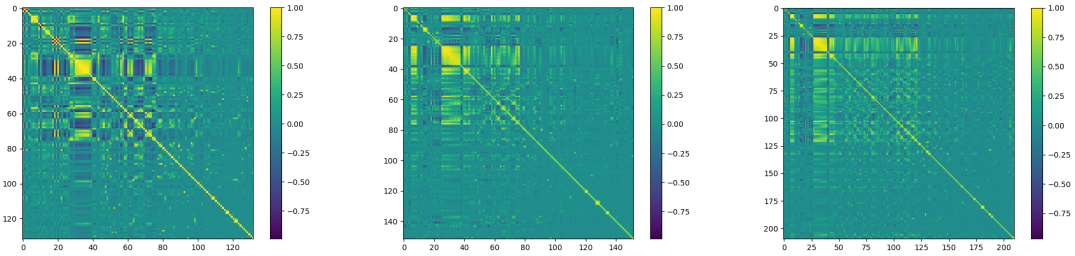
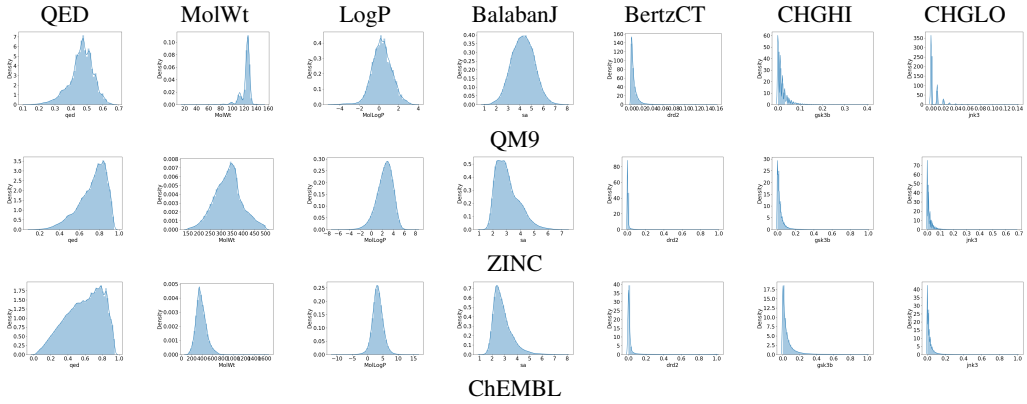


Figure 6: Inter-correlation heatmaps for studied molecular properties in QM9, ZINC and ChEMBL datasets.

Figure 7: Property distributions of 7 randomly selected molecular properties on QM9, ZINC and ChEMBL datasets.



C Latent Space Evaluation

To evaluate the quality of the learned latent space, we utilize three disentanglement evaluation metrics, disentanglement, completeness and informativeness [58]. To be specific, disentanglement measures the degree to which each latent dimension controls at most one molecular property, completeness measures the degree to which each molecular property is governed by at most one latent dimension, and informativeness measures the prediction accuracy of molecular properties given the latent representation. From Table 4, we find MoFlow learns a better and more disentangled latent space than CGVAE and HierVAE. One possible reason is that MoFlow (369) has a larger latent space than CGVAE (100) and HierVAE (32) since Flow restricts the latent size to be equal to the input size. Similarly, CGVAE ranks the second likely because its latent space size is larger than HierVAE.

D Molecule Manipulation Experiments

D.1 Molecule Genration Evaluation

We evaluate the **Validity**, **Novelty** and **Uniqueness** of the generated molecules as proposed in Kusner et al. [59] in Table 5. We can observe that ChemSpacE not only improves the success rate from the baseline methods, but also in general improves the novelty and uniqueness during manipulation.

Besides, in Fig. 8, we also report the SSR curves of molecule manipulations over three models on QM9 and ZINC datasets with multiple manipulation ranges (distance in the latent space), $[-1, 1]$, $[-5, 5]$, $[-10, 10]$ and $[-20, 20]$. From the figure, we can observe that the trends in each of the curves remain still when the manipulation range changes. In general, either too large or too small range is not desired, we set it as a hyper-parameter and we observe that $[-1, 1]$ is a reasonably good default value. More experiments on molecule manipulation can be found in Appendix D.

Table 4: Quantitative Evaluation of Disentanglement on Latent Space.

Datasets	Models	Disentanglement	Compleness	Informativeness
QM9	MoFlow	**0.24**	**0.57**	**0.83**
	HierVAE	0.13	0.27	0.75
ZINC	MoFlow	**0.40**	**0.62**	0.87
ChEMBL	HierVAE	**0.14**	**0.41**	**0.81**

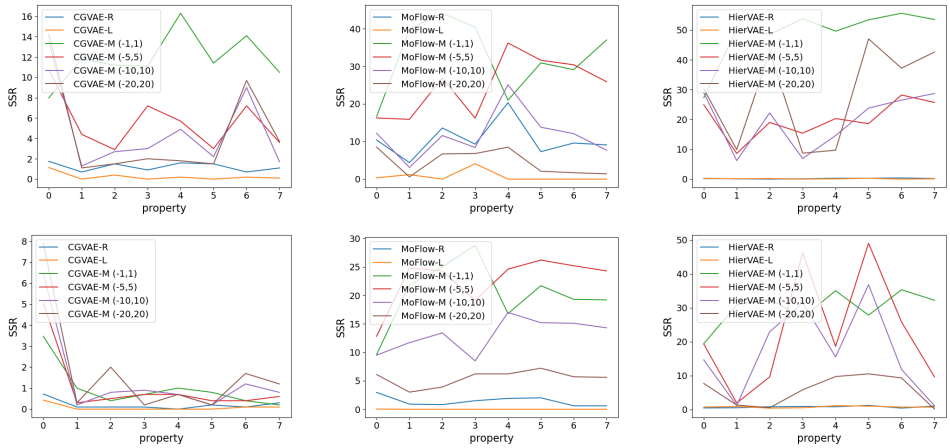


Figure 8: Molecule manipulation performance (average) with various manipulation ranges with three models on QM9 (top) and ZINC (bottom) datasets.

D.2 Molecule Manipulation Evaluation

In this section, we report detailed results for all manipulation ranges $[-1, 1]$, $[-5, 5]$, $[-10, 10]$, $[-20, 20]$ in terms of success rate and strict success rate in Table 6. Additionally, we visualize the SSR curves of molecule manipulations over three models on QM9 and ZINC in Fig. 9 and SR/SSR

Table 5: Quantitative Evaluation of Latent Manipulation.

Datasets	Models	Validity (%)	Novelty (%)	Uniqueness (%)
QM9	MoFlow	100.00	98.23	98.27
	MoFlow-R	91.60	91.60	8.06
	MoFlow-L	40.75	40.75	9.32
	MoFlow-C	91.63	88.71	24.23
QM9	HierVAE	100.00	79.39	95.14
	HierVAE-R	100.00	84.53	28.91
	HierVAE-L	100.00	84.05	27.26
	HierVAE-C	100.00	79.66	34.81
ZINC	MoFlow	100.00	100.00	100.00
	MoFlow-R	69.98	69.98	29.04
	MoFlow-L	43.36	43.36	24.87
	MoFlow-C	71.26	71.26	15.82
ChEMBL	HierVAE	100.00	94.03	99.45
	HierVAE-R	100.00	84.53	28.91
	HierVAE-L	100.00	93.00	55.09
	HierVAE-C	100.00	94.24	56.58

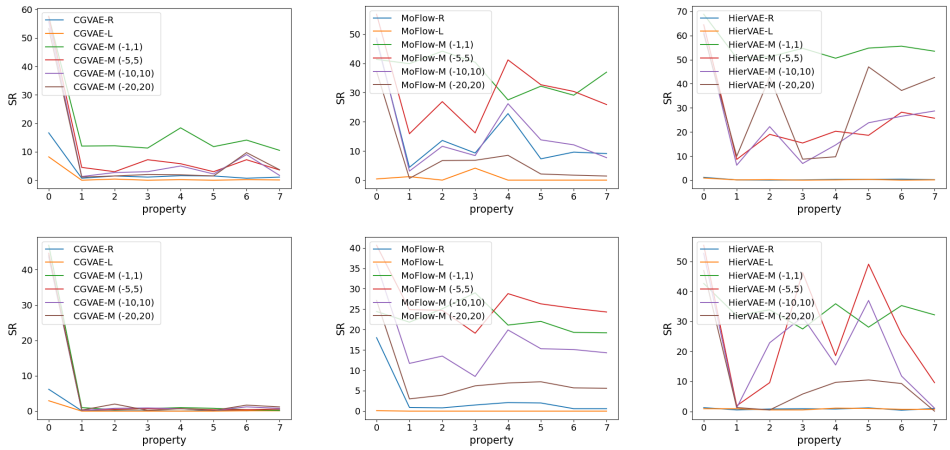


Figure 9: Molecule manipulation performance with various manipulation ranges with three models on QM9 (top) and ZINC (bottom) datasets (better seen in color).

Table 6: Quantitative Evaluation of Molecule Manipulation over a variety of molecular properties (numbers reported are *soft success rate-L* / *soft success rate-G* in %, -R denotes model with random manipulation, -L denotes model with largest range manipulation, -O denotes optimization-based manipulation, -C denotes model with ChemSpacE. The best performances are bold.

Datasets	Models	Avg.	QED	LogP	SA	DRD2	JNK3	GSK3B	MolWt
QM9	MoFlow-R	27.21 / 32.31	1.50 / 2.00	0.00 / 3.00	1.00 / 3.00	0.00 / 46.00	4.00 / 4.00	0.00 / 15.50	1.50 / 55.00
	MoFlow-L	29.28 / 35.20	3.00 / 8.00	1.00 / 7.00	1.00 / 2.00	0.50 / 42.50	6.00 / 6.00	0.50 / 7.50	1.00 / 58.00
	MoFlow-O	N/A	4.50/6.50	6.50/8.50	8.50/13.00	3.00/15.0	10.50/10.50	10.50/17.50	8.50/22.00
	MoFlow-C	**53.97 / 61.56**	**16.00 / 28.00**	**13.50 / 28.00**	**17.50 / 39.50**	**17.50 / 72.50**	**58.50 / 58.50**	**21.50 / 49.00**	**15.00 / 72.00**
	HierVAE-R	2.62 / 26.06	1.00 / 1.00	1.50 / 1.50	0.50 / 0.50	0.50 / 1.50	1.00 / 5.50	1.00 / 3.00	0.50 / 2.50
	HierVAE-L	3.25 / 27.33	0.50 / 1.00	0.00 / 1.50	0.00 / 5.50	0.50 / 4.00	2.00 / 8.50	0.00 / 2.50	0.50 / 1.50
	HierVAE-C	**46.72 / 61.49**	**27.00 / 35.00**	**32.00 / 44.00**	**35.00 / 42.00**	**41.50 / 48.50**	**51.50 / 60.00**	**30.00 / 33.50**	**39.50 / 45.50**
ZINC	MoFlow-R	35.85 / 41.70	3.50 / 6.00	2.50 / 7.50	3.50 / 6.50	5.50 / 79.00	4.00 / 56.50	1.50 / 27.50	4.50 / 12.50
	MoFlow-L	37.46 / 43.12	3.00 / 4.50	9.00 / 15.50	2.00 / 6.00	8.00 / 81.50	4.00 / 67.50	4.00 / 33.00	3.00 / 14.50
	MoFlow-O	N/A	1.50/2.00	10.50/15.50	1.00/2.50	2.50/5.50	18.00/21.50	23.50/28.50	0.50/1.50
	MoFlow-C	**60.54 / 63.23**	**53.50 / 57.00**	**57.00 / 73.50**	**54.00 / 61.50**	**55.50 / 65.50**	**57.50 / 63.50**	**56.00 / 68.00**	**56.00 / 71.00**
ChEMBL	HierVAE-R	0.24 / 18.20	0.00 / 0.00	0.00 / 0.50	0.00 / 0.50	0.00 / 2.00	0.00 / 0.00	0.00 / 1.00	0.00 / 0.00
	HierVAE-L	0.25 / 17.88	0.00 / 0.00	0.00 / 2.50	0.00 / 0.00	0.00 / 0.50	0.00 / 1.00	0.00 / 0.00	0.00 / 2.00
	HierVAE-C	**13.76 / 36.26**	**0.50 / 2.50**	**3.00 / 3.50**	**3.00 / 5.00**	**6.00 / 11.00**	**7.50 / 15.00**	**5.50 / 9.00**	**4.50 / 9.00**

curves of molecule manipulation with HierVAE on ChEMBL datasets in Fig. 10. The manipulation visualization of CGVAE on QED, MolWt and LogP is provided in Fig. **??**.

E ChemSpacE Demo

As shown in Fig. 11(right), we design an interactive real-time system for molecule manipulation, where the user can click random to randomly sample a molecule and freely select which model to interpret, which property to interpret, and tuning the slide bar manipulates the molecule accordingly in real-time. The demo video is anonymously provided at `https://drive.google.com/drive/folders/1N036p_5OfvGZybgPJ3Vw1ONXHVepimSR?usp=sharing`.

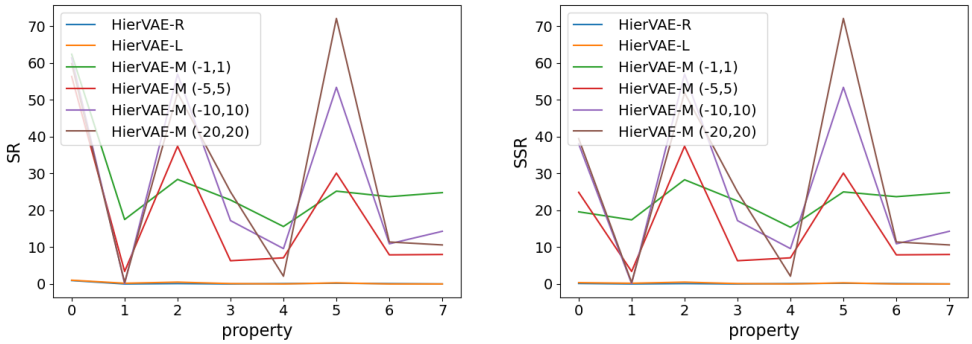


Figure 10: Molecule manipulation performance with various manipulation ranges with HierVAE on ChEMBL dataset (left SR, right SSR) (better seen in color).

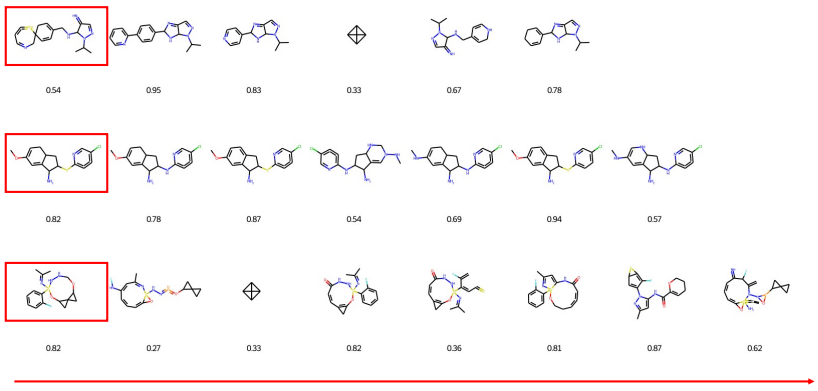


Figure 11: Optimizing molecular properties with optimization-based method.