ChemSpacE: Interpretable and Interactive Chemical Space Exploration

Anonymous authors
Paper under double-blind review

Abstract

Discovering meaningful molecules in the vast combinatorial chemical space has been a long-standing challenge in many fields from materials science to drug discovery. Recent advances in machine learning, especially generative models, have made remarkable progress and demonstrate considerable promise for automated molecule design. Nevertheless, most molecule generative models remain black-box systems, whose utility is limited by a lack of interpretability and human participation in the generation process. In this work we propose **Chem**ical **Spac**e **E**xplorer (ChemSpacE), a simple yet effective method for exploring the chemical space with pre-trained deep generative models. It enables users to interact with existing generative models and inform the molecule generation process. We demonstrate the efficacy of ChemSpacE on the molecule optimization task and the molecule manipulation task in single property and multi-property settings. On the molecule optimization task, the performance of ChemSpacE is on par with previous black-box optimization methods yet is considerably faster and more sample efficient. Furthermore, the interface from ChemSpacE facilitates human-in-the-loop chemical space exploration and interactive molecule design.

1 Introduction

Designing new molecules with desired properties is crucial for a wide range of tasks in drug discovery and materials science (Chen et al., 2018). Traditional pipelines require exhaustive human efforts and extensive domain knowledge to explore the vast combinatorial chemical space, making them difficult to scale up. Recent studies exploit deep generative models to tackle this problem by encoding molecules into a meaningful latent space, from which random samples are drawn and decoded to new molecules. Such deep molecule generative models can facilitate the design and development of drugs and materials (Lopez et al., 2020; Sanchez-Lengeling & Aspuru-Guzik, 2018).

Despite the promising results of deep generative models for molecule generation, considerably less effort has been made in understanding their underlying working mechanisms, which are key to interpretable and interactive AI-empowered molecule design. Most existing models are based on deep neural networks or black-box optimization methods, which lack transparency and interpretability (Samek et al., 2019). 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 (Zhou et al., 2016), model space (Guo et al., 2021), and latent space (Shen et al., 2020; Shen & Zhou, 2021). In the molecule generation domain, interpretability can be studied from two perspectives: (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** where adjusting molecular properties could observe smooth structure change of molecules.

Furthermore, it remains difficult to generate molecules with desired properties. Previous works tackle the problem with reinforcement learning-based, latent space optimization-based, and searching-based methods to achieve property control of the generated molecules (Shi et al., 2020; Jin et al., 2018a). Specifically, reinforcement learning-based algorithms (You et al., 2018a) equip the model with rewards designed to encourage the models to generate molecules with specific molecular properties. Latent space optimization-based algorithms take advantage of the learned latent space of molecule generative models and optimize the molecular properties via Bayesian Optimization (Liu et al., 2018). Searching-based algorithms directly search the discrete and high-dimensional chemical space for molecules with optimal properties (Kwon et al., 2021). However, these works often have three major issues. (1) They require many expensive oracle calls to provide feedback (*i.e.*, property scores) of the intermediate molecules during the searching or optimization process Huang et al. (2021). (2) They often only focus on the outcome of the process while ignoring its intermediate steps which can be essential for chemists and pharmacologists in understanding the chemical instances and rules that govern the process. (3) They stick to local gradients while putting less focus on global directions in the chemical/latent space.

To tackle the above challenges, we propose a simple yet effective method to explore the chemical space for molecule generation by leveraging the latent space of the pre-trained deep generative models. The motivation for our approach is based on the emergent properties of the latent space learned by molecule generative models Gómez-Bombarelli et al. (2018); Zang & Wang (2020): (1) molecules sharing similar structures/properties tend to cluster in the latent space, (2) interpolating two molecules in the latent space leads to smooth changes in molecular structures/properties. Thus, we develop *ChemSpace Explorer*, a model-agnostic method to manipulate molecules with smooth changes of molecular structures and properties which has broad applications ranging from molecule optimization to chemical space interpretation. 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, it then estimates the *latent directions* that govern molecular properties, which enable smooth change of molecular structures and properties without model re-training. This manipulation process improves the interpretability of deep generative models by navigating their latent spaces and enables *human-in-the-loop* exploration of the chemical space and molecule design. It allows users to manipulate the properties of generated molecules by leveraging the steerability and interpretability of molecule generative models. To the best of our knowledge, this work is the first attempt to achieve interactive molecule discovery by steering pre-trained molecule generative models.

Our experiments demonstrate that our method can efficiently and effectively steer state-of-the-art molecule generative models for molecule manipulation with a 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. In addition, we compare ChemSpacE with a gradient-based optimization method that traverses the latent space of molecule generative models on the molecule optimization task. To facilitate the interactive molecule design and discovery for practitioners, we further develop an interface for real-time interactive molecule manipulations and smooth molecular structure/property changes. We summarize the main contributions as follows:

- We explore a new task on *latent molecule manipulation*, which aims at steering the latent space of molecule generative models for manipulating the chemical properties of the output molecule and facilitating *human-in-the-loop* molecule design.
- We develop an efficient model-agnostic method named *ChemSpacE* for molecule manipulation, which can be incorporated in various pre-trained state-of-the-art molecule generative models without re-training or modification.
- We demonstrate the effectiveness and efficiency of our method in molecule optimization and achieving *human-in-the-loop* molecule design through comprehensive experiments. We further develop an interface to exhibit interactive molecule discovery and design.

2 Problem Formulation of Molecule Manipulation

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 latent space dimension, and further decodes latent vector back to the molecular space. Specifically, variational auto-encoder (VAE) (Kingma & Welling, 2013) and flow-based model (Flow) (Rezende & Mohamed, 2015) are the two most commonly used models for molecule generation, which typically encode the data from molecular space to latent space of Gaussian distribution. The encoding and decoding process can be formulated as:

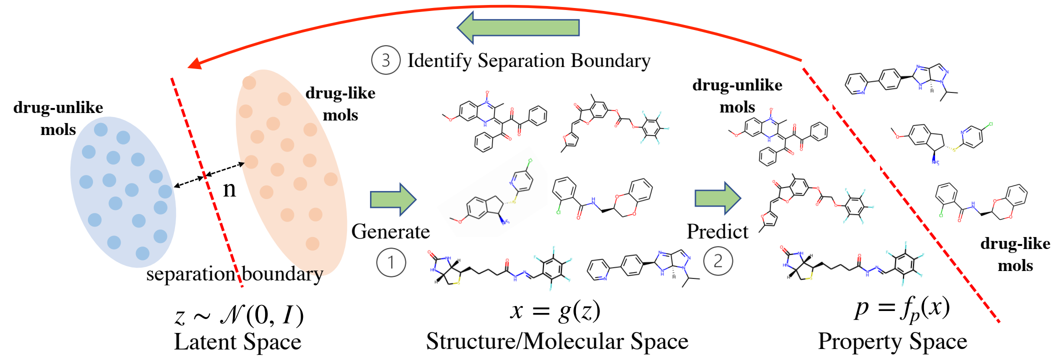


Figure 1: *ChemSpacE* framework: (1) the tested molecule generative model generates novel molecules by sampling random vector from the latent space and then feeding it into the generator, (2) off-the-shelf oracle function is used to predict molecular properties from the chemical space, (3) ChemSpacE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

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

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.

Molecule Manipulation Formulation. To leverage the steerability and interpretability of molecule generative models, we explore 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 $g\colon \mathcal{Z} \rightarrow \mathcal{X}$, where $\mathcal{Z} \in \mathbf{R}^l$ stands for the l-dimensional latent space, which is commonly assumed to be Gaussian distribution (Kingma & Welling, 2013; Rezende & Mohamed, 2015). There exist property functions f_P which define the property space $\mathcal{P}$ via $P = f_P(X)$. 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 $\mathcal{Z} \rightarrow \mathcal{X} \rightarrow \mathcal{P}$, where the relationship between $\mathcal{Z}$ and $\mathcal{X}$ explains the latent 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. While this criteria is rather strict, we propose an alternative relaxed success rate that tolerates a small deviation along the manipulation path, detailed in Appendix A.

3 ChemSpacE for Molecule Manipulation

3.1 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

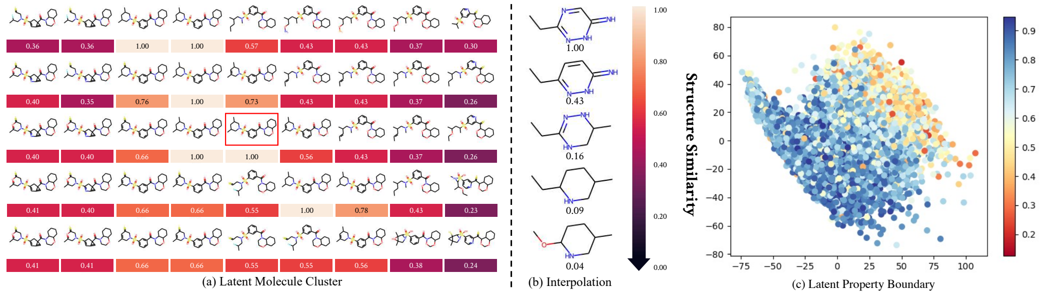


Figure 2: (a) Molecule clusters in the latent space, the number represents structure similarity (Bajusz et al., 2015), where the red box represents the base molecule, x and y axes denote two random orthogonal directions to manipulate. (b) Linear interpolation of two (top and bottom) molecules. (c) Latent property boundary is visualized for QED property.

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 (Seybold et al., 1987), 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 (Gómez-Bombarelli et al., 2018; Zang & Wang, 2020). 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.

3.2 Identifying Latent Directions

Latent Separation Boundary. With the verifications above and the previous work of analyzing the latent space of generative models (Shen et al., 2020; Bau et al., 2017; Jahanian et al., 2019; Plumerault et al., 2020), 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 the distance from any sample z to the separation boundary as:

$$d(z, n) = n^T z. \tag{2}$$

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:

$$f_P(g(z)) = \alpha \cdot d(z, n), \tag{3}$$

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:

$$f_P(g(z)) = AN^T z, \tag{4}$$

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:

$$\mu_P = \mathbf{E}(AN^T z) = AN^T \mathbf{E}(z) = \mathbf{0}, \tag{5}$$

$$\Sigma_P = \mathbf{E}(AN^T zz^T NA^T) = AN^T \mathbf{E}(zz^T) NA^T = AN^T NA^T. \tag{6}$$

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.

3.3 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

$$z' = z + \alpha n. \tag{7}$$

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

$$f_P(g(z + \alpha n)) = f_P(g(z)) + k\alpha. \tag{8}$$

For single-property manipulation, we can simply take the identified direction, but when multiple properties correlate with each other, we need to determine whether the two directions are entangled or disentangled. We can then simply take the disentangled and positively correlated attributions of the directions as the new direction:

$$n = n_1 + (1_{[n_1 \odot n_2 \geq 0]}) \odot n_2. \tag{9}$$

4 Experiments

4.1 Setup

Datasets. We use three molecule datasets, QM9 (Ramakrishnan et al., 2014), ZINC250K (Irwin & Shoichet, 2005), and ChEMBL (Mendez et al., 2019). QM9 contains 134k small organic molecules with up to 9 heavy atoms (C, O, N, F). ZINC250K (Gómez-Bombarelli et al., 2018) is a sampled 250K molecules from ZINC, a free database of commercially-available compounds for drug discovery with an average of ∼23 heavy atoms. ChEMBL is a manually curated database of bioactive molecules with drug-like properties and contains ∼1.8 million molecules.

Baselines. We include two baseline methods of identifying latent direction that governs the molecular property and one gradient-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. **Gradient-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 for HierVAE (Jin et al., 2020) and MoFlow (Zang & Wang, 2020), respectively. All the molecular properties are calculated by RDKit Landrum et al. (2013) and TDC Huang et al. (2021). We utilize the implementation of linear models (linear SVM) from Scikit-learn Pedregosa et al. (2011). More details are available in Appendix A.

Interactive Demo. An interactive demo for molecule manipulation is provided at https://drive.google.com/drive/folders/1N036p_5OfvGZybgPJ3Vw1ONXHVepimSR?usp=sharing and one example is shown in Fig. 4 (right).

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 gradient-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**

Table 2: Efficiency in terms of training/inference time, data, and number of oracles of ChemSpacE compared to the gradient-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×

4.2 Quantitative Evaluation of Molecule Manipulation

In Table 1, Table 5 (Appendix), Table 6 (Appendix), and Table 2, we report the quantitative evaluation results for both single property and multi-property 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 tables, we can obtain the following insights.

(1) Our proposed method, ChemSpacE, as the first attempt for molecule manipulation, achieves excellent performance in manipulating both single and multi-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 (Seybold et al., 1987). 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 gradient-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×, required data for at least 400×, and required oracle calls for at least 400×.

Table 3: Single property molecule optimization for Penalized-logP on ZINC dataset with four comparison methods (δ is the threshold for similarity between the optimized and base molecules).

	JT-VAE			GCPN			LIMO		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	1.91 ± 2.04	0.28 ± 0.15	97.5%	4.20 ± 1.28	0.32 ± 0.12	100%	10.1 ± 2.3	NA	100%
0.2	1.68 ± 1.85	0.33 ± 0.13	97.1%	4.12 ± 1.19	0.34 ± 0.11	100%	5.8 ± 2.6	NA	99.0%
0.4	0.84 ± 1.45	0.51 ± 0.10	83.6%	2.49 ± 1.30	0.48 ± 0.08	100%	3.6 ± 2.3	NA	93.7%
0.6	0.21 ± 0.71	0.69 ± 0.06	46.4%	0.79 ± 0.63	0.68 ± 0.08	100%	1.8 ± 2.0	NA	85.5%

	MoFlow			**ChemSpacE**		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	8.61 ± 5.44	0.30 ± 0.20	98.88%	9.94 ± 6.09	0.18 ± 0.14	100%
0.2	7.06 ± 5.04	0.43 ± 0.20	96.75%	7.17 ± 5.59	0.42 ± 0.21	96.00%
0.4	4.71 ± 4.55	0.61 ± 0.18	85.75%	4.16 ± 4.43	0.65 ± 0.20	84.38%
0.6	2.10 ± 2.86	0.79 ± 0.14	58.25%	1.76 ± 2.40	0.81 ± 0.15	59.63%

4.3 Quantitative Evaluation of Molecule Optimization

We further compare our methods under the common molecule optimization setting including two tasks *single property constrained optimization* and *multi-property constrained optimization*. Beginning with a set of candidate molecules, we aim to optimize the molecular properties while keeping the similarities of the optimized molecules to be as close to the base molecules as possible. The setting is persuasive in many drug discovery tasks where one needs to optimize the properties of a given molecule while keeping the structure similar.

Single Property Constrained Optimization. We follow and compare with four previous works Jin et al. (2018a); You et al. (2018a); Zang & Wang (2020); Eckmann et al. (2022) with the exact same set of molecules on penalized logP property and test four different similarity constraint thresholds, we report the property improvement and similarity compared to the base molecule as well as the percentage of successfully optimized molecule within the threshold in Table 3. For our reported result, ChemSpacE is manipulating over the latent space learned by MoFlow, as MoFlow leverages gradient-based method which traces the local gradient that leads to property improvement in every step while we take on a more efficient way to learn the global improvement direction and follow it for all steps, we are performing surprisingly well and even better than the gradient-based method used in MoFlow. This empirically supports our assumption about the latent space exploration.

Multi-Property Constrained Optimization. As this is not reported by previous work on molecule optimization, we propose to optimize QED and penalized logP as a multi-property constrained optimization task. We also propose two simple baselines: (1) we add up the two properties (QED and penalized logP) to be optimized as a new objective and runs single-property constrained optimization on it, (2) we take into account the two gradient directions on the two properties and each step we move to both directions for gradient ascent. As shown in Appendix (Table 7), we demonstrate the capability of ChemSpacE for efficient multi-objective optimization. Our method improves both QED and penalized logP more than the two gradient-based methods. We showcase two examples in Fig. 5 that demonstrates ChemSpacE can optimize molecules with high structure perseverance and desired properties.

4.4 Qualitative Evaluation of Molecule Manipulation and Interpretation

In Fig. 6, 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.

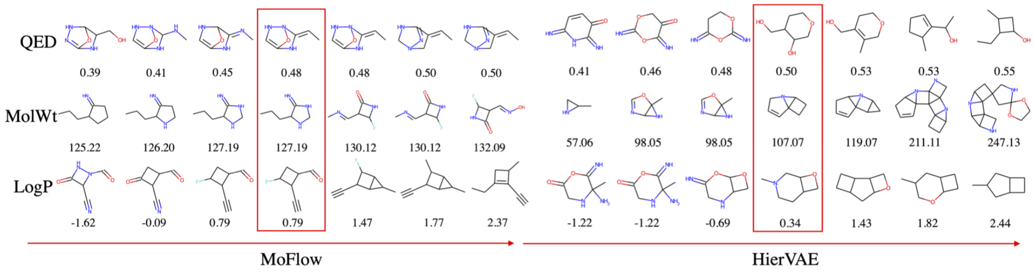
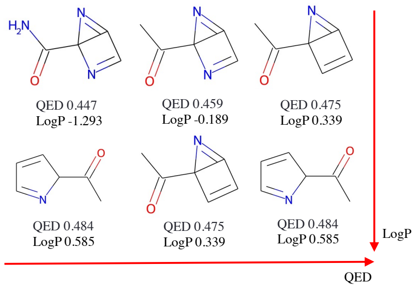
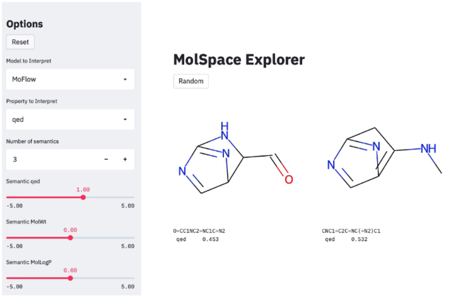


Figure 3: Manipulating QED, MolWt and LogP properties of sampled molecules. The backbone model is MoFlow and HierVAE trained on QM9 dataset.




Options

Reset

Model to Interpret: MoFlow

Property to Interpret: qed

Number of semantics: 3

Semantic qed: 1.00 (-5.00 to 5.00)

Semantic MolWt: 0.00 (-5.00 to 5.00)

Semantic MolLogP: 0.00 (-5.00 to 5.00)

MolSpace Explorer

Random

```
O=CC1NC2=NC1C=N2
 qed     0.453

CNC1=C2C=NC(=N2)C1
 qed     0.532
```

Figure 4: 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 D demo video for interactive molecule discovery (right).

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.

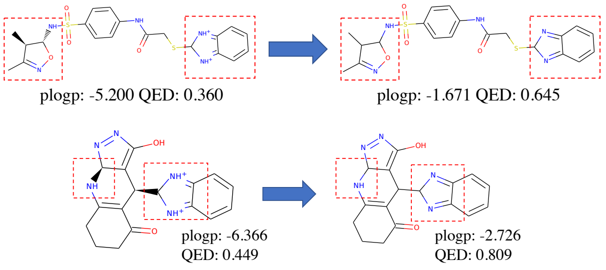


Figure 5: Illustrations of multi-property constrained optimization, the Tanimoto similarity between base and optimized molecules is 0.709 (top row) and 0.647 (bottom row) respectively.

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. 4 (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.

5 Related Work

Molecule Generation. Recent studies have explored a variety of deep generative models for molecule generation Du et al. (2022), such as variational autoencoders (VAEs) (Jin et al., 2018a), generative adversarial networks (GANs) (De Cao & Kipf, 2018), normalizing flows (Madhawa et al., 2019; Shi et al., 2020; Luo et al., 2021), energy-based models (EBMs) (Liu et al., 2021), reinforcement learning (Olivecrona et al., 2017; Zhou et al., 2019; Yang et al., 2021), *etc* (Yang et al., 2020; Xie et al., 2021). To be specific, JT-VAE (Jin et al., 2018a) proposes a VAE-based architecture to encode both atomic graphs and structural graphs for efficient molecule generation. MolGAN (De Cao & Kipf, 2018) exploits GANs for molecule generation, where discriminators are used to encourage the model to generate realistic and chemically-valid molecules. MRNN (Popova et al., 2019) extends the idea of GraphRNN (You et al., 2018b) to formulate molecule generation as an auto-regressive process. GCPN (You et al., 2018a) 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 (Madhawa et al., 2019) first introduces normalizing flows for molecule generation, where the generation process is invertible. Later works improve the flow-based models via auto-regressive generation (Shi et al., 2020), valency correction (Zang & Wang, 2020), and discrete latent representation (Luo et al., 2021). GraphEBM (Liu et al., 2021) 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 (Segler et al., 2018) enforces bias on the distribution of the data and trains the generative models with known desired properties to generate molecules with desired properties, while recent works mainly leverage latent space gradient-based (Jin et al., 2018a; You et al., 2018a; Hoffman et al., 2020; Winter et al., 2019), reinforcement learning-based (Shi et al., 2020; Zang & Wang, 2020; Blaschke et al., 2020), and searching-based (Brown et al., 2019; Yang et al., 2020; Kwon et al., 2021) approaches to generate molecules with desired properties. Latent space gradient-based methods are quite flexible and can work directly on both the molecules (Fu et al., 2022) and the learned latent vectors (Gómez-Bombarelli et al., 2018; Jin et al., 2018b; Winter et al., 2019; Griffiths & Hernández-Lobato, 2020; Notin et al., 2021). 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 (Brown et al., 2019; Yang et al., 2020; Kwon et al., 2021; Renz et al., 2019; Fu et al., 2020; Xie et al., 2021; Maziarz et al., 2021) are also capable of searching over chemical space for molecules with desired properties by defining a set of discrete actions. Besides, a few works (Chenthamarakshan et al., 2020; Das et al., 2021) leverage the learned latent space and achieve controllable generation by accepting/rejecting sampled molecules based on a molecular property predictor.

6 Conclusion, Limitation and Future Work

In this work, we develop a simple yet effective method called ChemSpacE to improve the steerability and interpretability of molecular generative models. The interface demonstrates the promising application of interactive molecule design and discovery. Nevertheless, we acknowledge two limitations of this work, (1) it cannot study the activity cliff phenomenon yet, (2) it lacks theoretical analyses about why the latent space of deep generative models is learned with property boundary. Specifically, we anticipate the enhanced understanding about the chemical space will lead to promising directions in understanding challenging phenomenons such as activity cliff — structurally similar molecules may have very different potency against the same target Stumpfe et al. (2014). However, activity cliff is a very challenging phenomenon which it requires specific benchmark datasets, and reliable oracle functions for molecule generation, thus is beyond the scope of this study. We leave this as a promising future work. Second, even though semantic direction in the latent space of generative models has been widely observed and leveraged, there have been few theoretical analyses which make it a challenging yet important question to answer in the future. We find this to be empirically meaningful and can be utilized to efficiently explore the chemical space.

References

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`.

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*, pp. 6541–6549, 2017.

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.

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.

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.

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.

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.

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

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.

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

Peter Eckmann, Kunyang Sun, Bo Zhao, Mudong Feng, Michael Gilson, and Rose Yu. Limo: Latent inceptionism for targeted molecule generation. In *International Conference on Machine Learning*, pp. 5777–5792. PMLR, 2022.

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, pp. 638–645, 2020.

Tianfan Fu, Wenhao Gao, Cao Xiao, Jacob Yasonik, Connor W Coley, and Jimeng Sun. Differentiable scaffolding tree for molecule optimization. In *International Conference on Learning Representations*, 2022.

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.

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.

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*, pp. 505–515, 2021.

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.

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.

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.

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

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, pp. 2323–2332. PMLR, 2018a.

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

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

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

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

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.

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

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

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

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

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

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

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.

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.

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.

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.

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.

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

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

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.

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.

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

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.

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

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.

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`.

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*, pp. 1532–1540, 2021.

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.

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.

Dagmar Stumpfe, Ye Hu, Dilyana Dimova, and Jurgen Bajorath. Recent progress in understanding activity cliffs and their utility in medicinal chemistry: miniperspective. *Journal of medicinal chemistry*, 57(1):18–28, 2014.

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.

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.

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.

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.

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*, 2018a.

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*, pp. 5708–5717. PMLR, 2018b.

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*, pp. 617–626, 2020.

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*, pp. 2921–2929, 2016.

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.

Appendix for “ChemSpacE: Interpretable and Interactive Chemical Space Exploration”

A Experiment 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 (Jin et al., 2020) embeds molecular structure motifs into a hierarchical VAE-based generative model; MoFlow (Zang & Wang, 2020) 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 Landrum et al. (2013) and protein binding affinity, synthesis accessibility oracles in TDC Huang et al. (2021). In total, we analyze 212 molecular properties from multiple dimensions, including distributions, inter-correlations, etc. Details can be found in Appendix G. 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. 2). 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.

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:

$$\phi_{SPC}(x,k,f) = 1[\forall\ i \in [k], s.t., f(x^{(i)}) - f(x^{(i+1)}) \leq 0], \tag{10}$$

$$\phi_{SSC}(x,k,\delta) = 1[\forall\ i \in [k], s.t., \delta(x^{(i+1)}, x^{(1)}) - \delta(x^{(i)}, x^{(1)}) \leq 0], \tag{11}$$

$$\phi_{DIV}(x,k) = 1[\exists\ i \in [k], s.t., x^{(i)} \neq x^{(1)}], \tag{12}$$

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:

$$SSR - L(P,X,k) = \frac{1}{|P| \times |X|} \sum\nolimits_{p \in P, x \in X} 1[\phi_{SPC}(x_p,k,f_p) \wedge \phi_{SSC}(x_p,k) \wedge \phi_{DIV}(x_p,k)], \tag{13}$$

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

$$\phi_{RPC}(x,k,f,\epsilon) = 1[\forall\ i \in [k], s.t., f(x^{(i)}) - f(x^{(i+1)}) \leq \epsilon], \tag{14}$$

$$\phi_{RSC}(x,k,\delta,\gamma) = 1[\forall\ i \in [k], s.t., \delta(x^{(i+1)}, x^{(1)}) - \delta(x^{(i)}, x^{(1)}) \leq \gamma], \tag{15}$$

$$\phi_{DIV}(x,k) = 1[\exists\ i \in [k], s.t., x^{(i)} \neq x^{(1)}], \tag{16}$$

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:

$$RSR-L(P, X, k, \epsilon_l, \gamma) = \frac{1}{|P| \times |X|} \sum\nolimits_{p \in P, x \in X}$$
$$\mathbb{1}[\phi_{RPC}(x_p, k, f_p, \epsilon_l) \wedge \phi_{RSC}(x_p, k, \gamma) \wedge \phi_{DIV}(x_p, k)], \tag{17}$$

$$RSR-G(P, X, k, \epsilon_g, \gamma) = \frac{1}{|P| \times |X|} \sum\nolimits_{p \in P, x \in X}$$
$$\mathbb{1}[\phi_{RPC}(x_p, k, f_p, \epsilon_g) \wedge \phi_{RSC}(x_p, k, \gamma) \wedge \phi_{DIV}(x_p, k)], \tag{18}$$

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).

Hyperparameters. ChemSpacE does not entail many hyperparameters, the only important one is the manipulation range which is critical to the exploration degree of the latent space. For molecule manipulation experiments, as we would like a gradual change over the molecular structure and property, we set the range as $[-1, 1]$. While for molecule optimization task, it requires more aggressive exploration strategies to reach the expected latent area which poses optimal property values. We utilize $[-100, 100]$ and $[-30, 30]$ for single property optimization and multi-property optimization experiments respectively. We report the results for single property optimization with ranges from $[1, 5, 10, 15, 20, 30, 50, 100]$ in Table 4.

B Extended Molecule Manipulation Experiments

B.1 Molecule Generation Evaluation

We evaluate the **Validity**, **Novelty** and **Uniqueness** of the generated molecules as proposed in Kusner et al. (2017) in Table 8. 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.

B.2 Molecule Manipulation Distribution Evaluation

We report the distribution shift of the properties during molecule manipulation in Fig. 6. Clearly, the property distributions shifts to the right when aiming to improve the molecular properties via identified directions and to the left when aiming to decrease the molecular properties via identified directions.

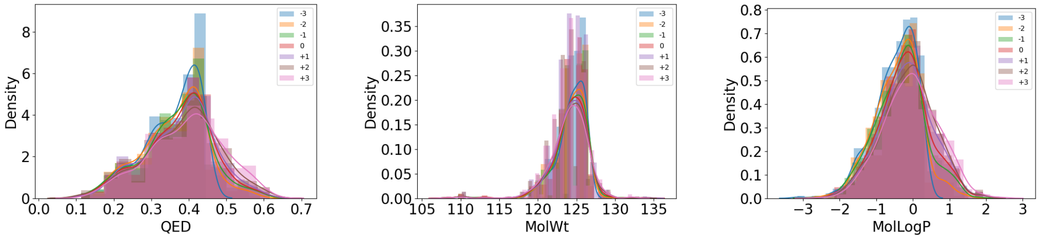


Figure 6: Visualization of Molecular property distribution shift while manipulating molecules with MoFlow on QM9 dataset (0 denotes the randomly sampled base molecule and $+x$ and $-x$ denote manipulation directions and steps).

B.3 Multi-property Molecule Manipulation Evaluation

We evaluate multi-property (penalized logp, QED) molecule manipulation over 200 randomly sampled molecules on ZINC dataset in Table 5.

Table 4: Single property molecule optimization for Penalized-logP on ZINC dataset with different manipulation ranges of ChemSpacE (δ is the threshold for similarity between the optimized and base molecules).

	ChemSpacE-1			ChemSpacE-5		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	2.61 ± 2.55	0.71 ± 0.23	83.25%	3.33 ± 3.74	0.67 ± 0.26	84.25%
0.2	2.56 ± 2.51	0.72 ± 0.22	97.1%	3.17 ± 3.60	0.69 ± 0.23	84.13%
0.4	2.26 ± 2.28	0.75 ± 0.20	77.25%	2.62 ± 3.08	0.73 ± 0.20	78.13%
0.6	1.34 ± 1.54	0.84 ± 0.14	57.0%	1.43 ± 1.54	0.84 ± 0.14	57.38%

	ChemSpacE-10			ChemSpacE-15		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	4.97 ± 4.86	0.57 ± 0.27	90.75%	5.92 ± 5.11	0.51 ± 0.26	93.75%
0.2	4.70 ± 4.71	0.60 ± 0.24	90.13%	5.62 ± 5.05	0.55 ± 0.23	93.25%
0.4	3.43 ± 3.96	0.69 ± 0.20	82.38%	3.96 ± 4.28	0.73 ± 0.20	84.25%
0.6	1.67 ± 2.32	0.82 ± 0.15	59.00%	1.73 ± 2.35	0.81 ± 0.15	59.63%

	ChemSpacE-20			ChemSpacE-30		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	6.62 ± 5.57	0.46 ± 0.25	94.40%	7.77 ± 6.34	0.39 ± 0.24	96.38%
0.2	6.11 ± 5.14	0.51 ± 0.22	93.75%	6.50 ± 5.40	0.48 ± 0.22	94.50%
0.4	4.22 ± 4.50	0.65 ± 0.19	85.13%	4.47 ± 4.73	0.64 ± 0.19	85.88%
0.6	1.79 ± 2.36	0.81 ± 0.15	59.88%	1.78 ± 2.37	0.81 ± 0.15	60.25%

	ChemSpacE-50			ChemSpacE-100		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	8.80 ± 6.35	0.30 ± 0.21	98.38%	9.94 ± 6.09	0.18 ± 0.14	100%
0.2	6.99 ± 5.53	0.44 ± 0.21	95.00%	7.17 ± 5.59	0.42 ± 0.21	96.00%
0.4	4.45 ± 4.65	0.63 ± 0.19	85.38%	4.16 ± 4.43	0.65 ± 0.20	84.38%
0.6	1.87 ± 2.56	0.80 ± 0.15	60.13%	1.76 ± 2.40	0.81 ± 0.15	59.63%

Table 5: Quantitative Evaluation of Molecule Manipulation for Multiple Properties. (-R denotes model with random manipulation, MoFlow-1 and MoFlow-2 denote two variants of gradient-based baseline methods, RSR(L) denotes *relaxed success rate-L*, RSR(G) denotes *relaxed success rate-G*).

Metric	MoFlow-1	MoFlow-2	**ChemSpacE**
SSR-both	28.00	27.00	**62.00**
RSR(L)-both	29.50	28.00	**63.00**
RSR(G)-both	41.00	38.50	**76.00**

Table 6: Quantitative Evaluation of Molecule Manipulation over a variety of molecular properties (numbers reported are *relaxed success rate-L / relaxed 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**

C Extended Molecule Optimization Experiments

We report more experiments about single property and multi-property optimization in this section. In Table 4, pushing further across the property separation boundary increases the improvement for molecule optimization but lowers the similarity scores.

D ChemSpacE Demo

As shown in Fig. 7(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`.

E Molecule Representations

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.

F Molecule Generative Models

In Table 9, 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 F that are very popular for molecule generation task.

Table 7: Multi-property molecule optimization for Penalized-logP and QED on ZINC dataset with two variants of gradient-based methods (δ is the threshold for similarity between the optimized and base molecules).

	MoFlow-1					
δ	**QED Improvement**	**QED % Improvement**	**pLogP Improvement**	**pLogP % Improvement**	**Similarity**	**Success**
0.0	0.17 ± 0.11	$42.06 \pm 35.69\%$	4.49 ± 3.87	$51.00 \pm 29.36\%$	0.44 ± 0.24	91.50%
0.2	0.16 ± 0.11	$37.84 \pm 32.16\%$	4.42 ± 3.78	$51.26 \pm 28.96\%$	0.48 ± 0.21	90.75%
0.4	0.12 ± 0.10	$29.53 \pm 27.45\%$	3.64 ± 3.43	$44.61 \pm 29.34\%$	0.61 ± 0.17	73.25%
0.6	0.07 ± 0.08	$17.44 \pm 20.36\%$	1.85 ± 2.18	$26.38 \pm 25.59\%$	0.78 ± 0.15	41.13%
	MoFlow-2					
δ	**QED Improvement**	**QED % Improvement**	**pLogP Improvement**	**pLogP % Improvement**	**Similarity**	**Success**
0.0	0.18 ± 0.12	$45.09 \pm 39.71\%$	4.67 ± 4.23	$50.74 \pm 28.79\%$	0.41 ± 0.23	92.88%
0.2	0.16 ± 0.11	$40.12 \pm 35.36\%$	4.48 ± 3.78	$51.32 \pm 29.11\%$	0.47 ± 0.20	91.50%
0.4	0.13 ± 0.10	$31.25 \pm 29.87\%$	3.70 ± 3.37	$45.16 \pm 29.27\%$	0.60 ± 0.17	74.88%
0.6	0.07 ± 0.08	$17.61 \pm 20.88\%$	1.97 ± 2.51	$26.74 \pm 26.30\%$	0.78 ± 0.15	41.88%
	ChemSpacE					
δ	**QED Improvement**	**QED % Improvement**	**pLogP Improvement**	**pLogP % Improvement**	**Similarity**	**Success**
0.0	0.20 ± 0.12	$50.75 \pm 41.77\%$	4.66 ± 4.34	$50.01 \pm 24.36\%$	0.34 ± 0.23	76.38%
0.2	0.18 ± 0.11	$42.70 \pm 32.87\%$	4.36 ± 3.50	$51.57 \pm 28.27\%$	0.45 ± 0.19	76.75%
0.4	0.14 ± 0.10	$33.59 \pm 27.92\%$	3.78 ± 3.49	$46.07 \pm 28.09\%$	0.58 ± 0.16	63.13%
0.6	0.08 ± 0.08	$20.12 \pm 22.33\%$	1.80 ± 1.81	$26.77 \pm 24.75\%$	0.77 ± 0.15	32.13%

Table 8: 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

F.1 Molecule Generative Model Formulation

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

$$\begin{aligned} \log p(x) &= \log \int_z p(x|z)p(z)dz \\ &\geq \log[\mathbb{E}_{q(z|x)}[p(x|z)] + \text{KL}(q(z|x)||p(z))] \end{aligned} \tag{19}$$

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

$$\begin{aligned} & z_0 \sim p_0(z_0) \\ & z_i = f_i(z_{i-1}) \\ & z_{i-1} = f_i^{-1}(z_i) \\ & p_i(z_i) = p_{i-1}(z_{i-1})\Big|\det\frac{df_i^{-1}}{dz_i}\Big| = p_{i-1}(f_i^{-1}(z_i))\Big|\det\frac{df_i^{-1}}{dz_i}\Big|, \end{aligned} \tag{20}$$

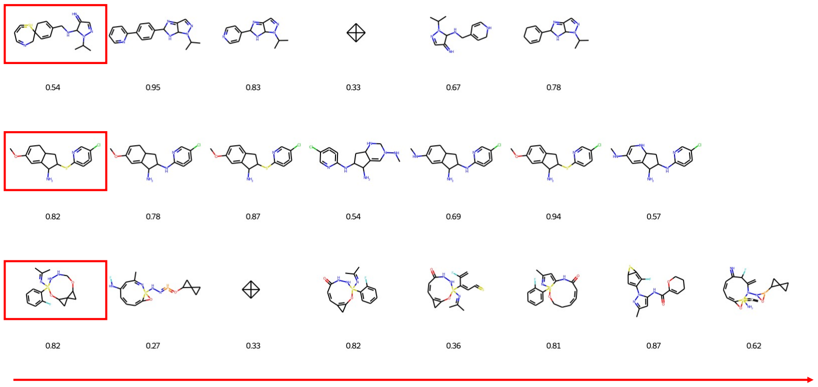


Figure 7: Optimizing molecular properties with optimization-based method.

Table 9: A summary of mainstream molecule generative models.

Prior Work	Generative Model	Sequential	Latent Space
JT-VAE (Jin et al., 2018a)	VAE	✓	✓
CGVAE (Liu et al., 2018)	VAE	✓	✓
MRNN (Popova et al., 2019)	RNN	✓	
GraphNVP (Madhawa et al., 2019)	Flow		✓
GCPN (You et al., 2018a)	RL	✓	
GraphAF (Shi et al., 2020)	Flow	✓	
MoFlow (Zang & Wang, 2020)	Flow		✓
HierVAE (Jin et al., 2020)	VAE	✓	✓
GraphEBM (Liu et al., 2021)	EBM		
GraphDF (Luo et al., 2021)	Flow	✓	

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:

$$\begin{aligned}
\log p(x) &= \log p_K(z_K) \\
&= \log p_{K-1}(z_{K-1}) - \log \left| \det \frac{df_K}{dz_{K-1}} \right| \\
&= \log p_{K-2}(z_{K-2}) - \log \left| \det \frac{df_{K-1}}{dz_{K-2}} \right| - \log \left| \det \frac{df_K}{dz_{K-1}} \right| \\
&= ... \\
&= \log p_0(z_0) - \sum_{i=1}^{K} \log \left| \det \frac{df_i}{dz_{i-1}} \right|
\end{aligned} \tag{21}$$

G 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,

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

[2] https://tdcommons.ai/

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. 8, 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.

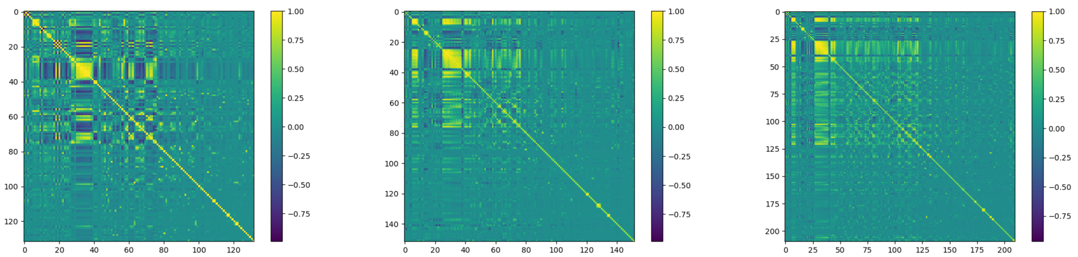
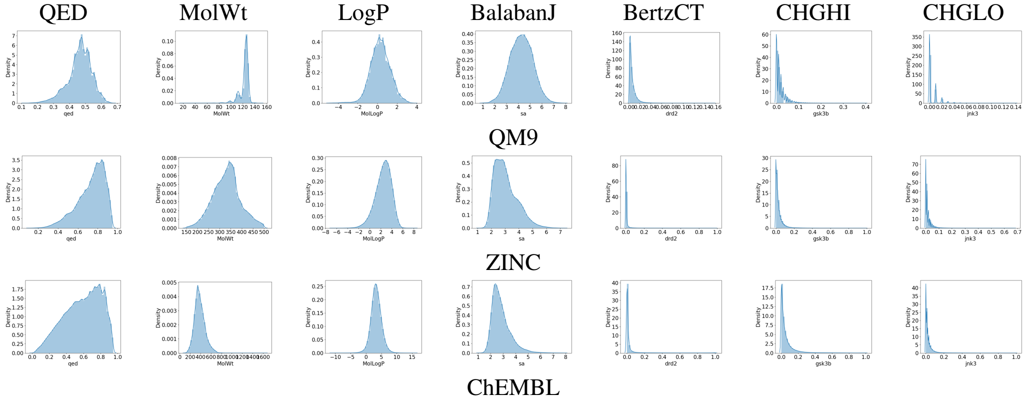


Figure 8: Inter-correlation heatmaps for studied molecular properties in QM9, ZINC and ChEMBL datasets.

Molecular Property Distributions. We visualize 7 molecular property distributions reported in section 4 in Fig. 9. 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.

Figure 9: Property distributions of 7 randomly selected molecular properties on QM9, ZINC and ChEMBL datasets.



H Latent Space Evaluation

To evaluate the quality of the learned latent space, we utilize three disentanglement evaluation metrics, disentanglement, completeness and informativeness (Eastwood & Williams, 2018). 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 10, we find MoFlow learns a better and more disentangled latent space than HierVAE. One possible reason is that MoFlow (369) has a larger latent space than HierVAE (32) since Flow restricts the latent size to be equal to the input size.

Table 10: Quantitative Evaluation of Disentanglement on Latent Space.

Datasets	Models	Disentanglement	Completeness	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**