APPLICATION OF GENERATIVE MODELS ON MODELING BIOLOGICAL MOLECULES

DEMOKAN ÇOBAN

Thesis supervisor: ALFREDO VELLIDO ALCACENA (Department of Computer Science)

Thesis co-supervisor: CAROLINE LEONORE KONIG

Degree: Master Degree in Artificial Intelligence

Thesis report

School of Engineering
Universitat Rovira i Virgili (URV)

Faculty of Mathematics
Universitat de Barcelona (UB)

Barcelona School of informatics (FIB)
Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

16/05/2023

I dedicate this thesis work to my family: I cannot thank you enough for your non-ending support and for believing in me at the most difficult times.

Acknowledgements

I am deeply indebted to my thesis Co-supervisor Prof. Caroline Leonore König for her patience and understanding towards me and to my Supervisor Prof. Alfredo Vellido Alcacena for his guidance and his time.

Abstract

The last decade has been the stage for many groundbreaking Artificial Intelligence technologies, such as revolutionary language models: Generative models capable of synthesizing surprisingly unique data. Such a novelty also brings about public concerns, primarily due to state-of-the-art models' "black box" nature. One of the domains that has quickly adopted the generative deep learning paradigm is drug discovery, which, from a pharmaceutical industry point of view, is an extremely expensive and time-consuming process. However, the inner workings of such models are not inherently understandable by humans, causing hesitation to fully trust their results. The concept of disentanglement is one of the fundamental requirements to explain generative models, determining the extent to which steerability and navigation can be achieved in the latent space. Unfortunately, the application potential of interpretability approaches has some limitations depending on the availability of generative latent factors. This work aims to shed some light on the synthesized latent spaces of state-of-the-art molecular generative models: A couple of basic assumptions made about the latent space characteristics are analyzed and potential pitfalls related to domain, architecture, and molecule representation preferences are addressed. The degree to which the steerability in the latent space is achieved is quantified by implementing a novel interpretability approach, providing the basis for the comparison of alternative model configurations. The experiments further revealed that modeling decisions have a direct impact on achievable interpretability; albeit limited by the intricacies of the medicinal chemistry domain.

Contents

List of Figures

Chapter 1

Introduction

Artificial Intelligence (AI) is continuously proving its potential in a wide range of tasks, such as in the recent development of large language models that can generate human-like dialogue, or image and video generative models capable of producing content that is almost indistinguishable from a perceptual viewpoint; technological feats that also bring along unavoidable public controversy. The last decade has been extremely fruitful for AI research, partly because of the advancement of computational power and the growing interest in the field that comes with the breakthroughs provided by Deep Learning (DL) approaches. Although DL models can arguably achieve superhuman performance in a wide spectrum of specific tasks, the black-box nature that comes with their exceeding complexity generates concerns about their adaptation into various domains.

Generative DL models have been on the stage from as early as the 1960s [Weizenbaum (1966)], though the ground-breaking architectures such as Generative Adversarial Networks (GAN)[Goodfellow et al. (2014)], Variational Autoencoders (VAE) [Kingma and Welling (2022)] and normalizing flow-based generative models [Dinh et al. (2015)] such as Glow [Kingma and Dhariwal (2018)] were only introduced in the last decade, paving the way for the current state-of-the-art architectures. A typical generative model maintains a latent space, which is a space of lower dimensionality than that of the input data, and maps the input data into this synthesized space. The space possesses certain properties that distinguish it from a mere encoding that enables the model to obtain meaningful decodings when sampling from the space. Either by implicitly or explicitly learning the statistical distribution of the input data, never-seen data can be generated by this type of models. On the negative side, however, their training be a challenging task. GANs can suffer from mode collapse, a phenomenon that occurs when the generator part of the network tends to generate the same data, in turn causing the discriminator to condition itself to always reject that output. Although there have been improvements in the loss function [Arjovsky et al. (2017)], the training phase can still be brittle. On the other hand, the VAE architecture is known to suffer from a concept called posterior collapse [Yacoby et al. (2022)], resulting in the model to learn uninformative latent encodings.

Within the large span of application domains, the implementation of generative models in drug discovery has led to substantial success in problems of inverse molecular design, also known as *de novo* molecular design. Virtual screening (VS) is the traditional approach to developing drugs from known molecules, which is a process that is extremely time-consuming and expensive. Developing a new drug can cost as much as 2.6 billion$ to pharmaceutical companies [Avorn (2015)]. Moreover, experts estimate the space of chemically valid molecules to be in the range of 10^{23} to 10^{60} [Polishchuk et al. (2013)][Reymond and Awale (2012)], rendering its complete exploration intractable. The VS-based drug design paradigm appears to be limited due to the constraint of operating on the basis of already known molecules. For example, 49% of small-molecule cancer drugs were based on natural products or their derivatives up until the year 2014 [Newman and Cragg (2016)][Sanchez-Lengeling and Aspuru-Guzik (2018)]. However, *de novo* drug design potentially enables practitioners to traverse the vast chemical space more effectively [Meyers et al. (2021)], therefore unlocking a great amount of potentially useful molecules to be used as a drug with desired properties. In 2020 [Stokes et al. (2020)], a message-passing network-based model was able to identify Halicin as a potent antibiotic, which was undetected with the traditional VS paradigm, due to Halicilin being structurally divergent from known antibiotics. Also, facilitating the drug development process by utilizing generative models may potentially translate into drugs being more accessible and cheaper. Moreover, a proper understanding of the latent space of the generative models will help practitioners to develop an intuition about the inner workings of the black-box models, therefore increasing the incorporation of such models into drug discovery tasks.

Unlike in the image domain, interpretability methods for drug generative models are of limited availability. Due to the intricacies of the domain, most of the work exists in a case-based format [Du et al. (2022)]. Moreover, efforts made for generative model explainability do not translate well into drug generative models, since ground truth latent generative factors are unknown for the molecules. The concept of disentanglement [Zeng et al. (2022)][Du et al. (2022)] is the main approach that most of the interpretability efforts revolve around. To counteract such shortcomings, some proposals have been made that aim to shed some light on the black-box models: ChemSpacE [Du et al. (2023)] is a novel approach to identifying hyperplanes that separate certain properties of the molecules in the latent space. Once discovered, such planes enable practitioners to steer in the model's latent direction.

In the DL field, certain datasets are designed to serve as a measuring stick for model performance on a specific task. For example, ImageNet [Deng et al. (2009)] is the gold standard dataset for the development of object recognition models. In a similar sense, MOSES [Polykovskiy et al. (2020)] is a benchmark dataset that is proposed to objectively assess drug generative models' distribution learning capabilities. Most of the work in the *de novo* drug design field focuses on the molecular property optimization aspect, therefore MOSES is an important benchmark to standardize generative model performance for the assessment of distribution learning capabilities. Within the framework of the benchmark, several models with different architectures have been proposed to serve as a baseline.

The goal of this thesis is to implement state-of-the-art molecule generative networks and attempt to gain insights into the latent space synthesized by different generative paradigms. The models are trained on the MOSES benchmark dataset and their generative performances are evaluated on molecule validity, uniqueness, and novelty metrics. In order to assess the disentanglement aspect of the generative models, a couple of basic assumptions over the latent space will be tested, then a state-of-the-art latent space explorer method will be implemented to probe into the generative processes of the *de novo* drug designer architectures. Later, analysis and assessment of domain, representation, and architecture-based difficulties will be laid out to address the issues faced when attempts are made to steer into meaningful directions in the latent space.

Chapter 2

State of the Art

2.1 Basic Concepts

There are various architectures available for the task of molecule generation. Methods vary depending on many factors such as the downstream task, the choice of molecule representation, and the complexity of the molecules being worked on. An architecture that is capable of working with relatively small molecules may fail to model more complex molecules.

Generative Models incorporate multiple DL concepts to process the molecular data. Depending on the preferred architecture, certain types of Recurrent Neural Networks (RNN), such as Long Short Term Memory (LSTM) or Gated Recurrent Units (GRU) are commonly incorporated into the typical encoding and decoding processes. Also, RNNs are being utilized by message-passing neural networks to update node representations; therefore their application is not restricted to string-based molecular representation. There are various applications where RNN-based architectures are used as a stand-alone molecule generative architecture, but more importantly, they are usually ingrained into many architectures, such as VAE and GAN to process molecular sequence information intermediates.

2.1.1 Recurrent Neural Networks (RNN)

Recurrent Neural Networks [Schmidt (2019)] are typically the preferred neural network architectures when working with sequence-natured data, which may come in various modalities: sound, text, genomes, or time series, to name a few. One of the major differences between RNNs and classic Multi-Layer Perceptrons (MLP) is that the latter do not contain any sort of cycles when the data is being propagated,

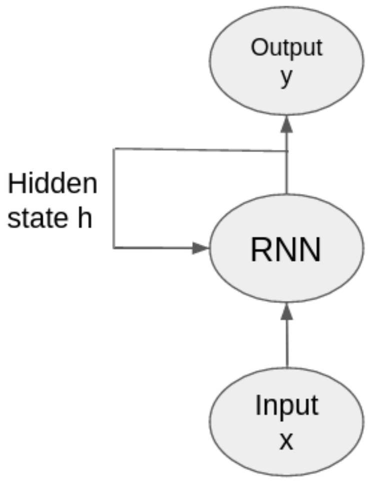


Figure 2.1: Basic diagram of the RNN architecture.

whereas in the former the network maintains information about the past states. To calculate input $\mathbf{X}_t$, all the past information $\mathbf{X}_{0:t-1}$ will be taken into consideration.

In order to describe the process mathematically, we can represent the hidden state of the network at time step t as $\mathbf{H}_t \in \mathbb{R}^{n\times h}$ and input at time step t as $\mathbf{X}_t \in \mathbb{R}^{n\times d}$ where n denotes the number of data points, d denotes the input dimensionality and h denotes the number of hidden units. After defining the weight matrix as $\mathbf{W}_{xh} \in \mathbb{R}^{d\times h}$, hidden state-to-state matrix as $\mathbf{W}_{hh} \in \mathbb{R}^{h\times h}$ and the bias parameter as $\mathbf{b}_h \in \mathbb{R}^{1\times h}$, we can obtain $\mathbf{H}_t$ after passing all the information from the activation function ϕ, which is usually a tanh or a logistic sigmoid function:

$$\mathbf{H}_t = \phi_h\left(\mathbf{X}_t\mathbf{W}_{xh} + \mathbf{H}_{t-1}\mathbf{w}_{hh} + \mathbf{b}_h\right) \tag{2.1}$$

Finally, the output $\mathbf{O}_t$ variable can be obtained by:

$$\mathbf{O}_t = \phi_o\left(\mathbf{H}_t\mathbf{w}_{ho} + \mathbf{b}_o\right) \tag{2.2}$$

The diagram 2.1 shows the typical architecture of a basic RNN.

In practice, standard RNNs struggle with long input sequences due to the vanishing gradient problem [Bian and Xie (2021)]. Certain improvements have been proposed to tackle the issues related to long dependencies: the most popular methods are LSTM and GRU Networks. By introducing a carry track to propagate information across the learning process, the signal vanishing issue is mitigated. LSTMs have a couple of gated cells: the Output Gate $\mathbf{O}_t$ that reads the entries of the cell;

the Forget Gate $\mathbf{F}_t$ that clears the information of the cell and $\mathbf{I}_t$ that reads the information into the cell. The dynamics can be expressed as:

$$\mathbf{O}_t = \sigma\left(\mathbf{X}_t\mathbf{W}_{xo} + \mathbf{H}_{t-1}\mathbf{w}_{ho} + \mathbf{b}_o\right) \tag{2.3}$$

$$\mathbf{I}_t = \sigma\left(\mathbf{X}_t\mathbf{W}_{xi} + \mathbf{H}_{t-1}\mathbf{w}_{hi} + \mathbf{b}_i\right) \tag{2.4}$$

$$\mathbf{F}_t = \sigma\left(\mathbf{X}_t\mathbf{W}_{xf} + \mathbf{H}_{t-1}\mathbf{w}_{hf} + \mathbf{b}_f\right) \tag{2.5}$$

The function σ is a sigmoid activation function that scales the raw output in the range $\in (0, 1)$.

Further, a new memory cell should be introduced that replaces the previous σ function with a $\tanh$ activation function that outputs in a range $\in (-1, 1)$.

$$\tilde{\mathbf{C}}_t = \tanh\left(\mathbf{X}_t\mathbf{W}_{xc} + \mathbf{H}_{t-1}\mathbf{w}_{hc} + \mathbf{b}_c\right) \tag{2.6}$$

By utilizing the information of the previously defined gates, we can dictate how much of the previous information will be maintained in the next state. When we combine the information as:

$$\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \tilde{\mathbf{C}}_t \tag{2.7}$$

Finally, by incorporating the hidden states $\mathbf{H}_t$ calculation, we can express the framework as:

$$\mathbf{H}_t = \mathbf{O}_t \odot \tanh(\mathbf{C}_t) \tag{2.8}$$

Gated Recurrent Units are also used as an alternative to LSTMs, although they are capable of holding shorter sequence information than LSTMs [Zeng et al. (2022)]. On the other hand, LSTMs can run into the risk of overfitting the data due to having a higher number of parameters. The choice will be depend on the size of the molecules under analysis.

2.1.2 Message Passing Neural Network (MPNN)

Message Passing Neural Networks (MPNN) [Gilmer et al. (2017)] are the basic backbone of how the graph information is processed by generative models. In each successive layer of the MPNN, a node's information will be aggregated with its neighborhood information along with the information of connectivity. In each successive pass, nodes start to get information about the neighbors-of-their neighbors. The forward pass consists of two phases: the message-passing phase and the readout phase. The former runs for T time steps and it is defined by message functions M_t and vertex update functions U_t. Throughout the message-passing phrase, hidden states h_v^t of each node is updated based on the messages m_v^{t+1} based on:

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{v,w}) \tag{2.9}$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{2.10}$$

where $N(v)$ is the neighbor nodes of node v in graph G. The readout phase calculates a feature vector for the graph as a whole via readout function R as:

$$\hat{y} = R(h_v^T \mid v \in G) \tag{2.11}$$

The functions of M_t, U_t and R are differentiable learned functions. The readout function works on node states and it is required for the function to be invariant to permutations, in turn enabling the MPNN to be invariant to graph isomorphism. Although many different variations exist, the basic operation is more or less the same across the alternatives.

2.1.3 Variational Autoencoders (VAE)

The Variational Autoencoder (VAE) [Kingma and Welling (2022)] is one of the main architectures utilized in the domain of generative modeling. VAE compresses and maps the input data as overlapping statistical distributions, typically Gaussian, as opposed to mere point encodings. This concept provides a couple of desired properties in the constructed latent space: encodings of the molecules are continuous, therefore, extrapolation between two elements will generate meaningful intermediates, resembling more to the point that is the closest. Also, similar items will be mapped to the neighborhood of each other. Moreover, by sampling from the synthesized latent space, new data can be generated.

Fundamentally, a VAE is an Autoencoder with a regularization aimed to encourage the model to map data as a distribution during the encoding and decoding process. In order to be decoded, the point will be sampled from the learned distribution and will be fed to the decoder. The encoder outputs the mean μ and the covariance matrix Σ for each input, which is encouraged by the loss function to typically be a standard Gaussian distribution. The mapping of data-point as distribution is enforced by the Kullback-Leiber (KL) divergence term in the loss function. The likelihood of observing data-point $\mathbf{x}$, expressed as $\mathbf{p}(\mathbf{x})$, is approximated by VAE [Du et al. (2022)]:

$$\begin{aligned}\log p(x) &= \log \int_z p(z)p(x \mid z)dz \\ &\geq \mathbb{E}_{q(z|x)}[\log p(x \mid z) - D_{KL}(q(z \mid x) \| p(z)) \\ &\triangleq \text{ELBO}\end{aligned} \tag{2.12}$$

The left-hand side of the ELBO equation 2.12 corresponds to the reconstruction loss, which is modeled as Gaussian distribution. The right-hand part of the equation, the KL divergence, quantifies the deviation between $p(z)$ and its approximation $q(z \mid x)$. The preference of standard Gaussian prior $p(z) \sim \mathcal{N}(0, I)$ supports disentanglement between the latent space dimensions, although by itself it is not sufficient to satisfy the disentanglement condition. In order to back-propagate the gradients, the VAE utilizes a re-parameterization trick to latent vector z by adding a random standard normal vector ϵ to the standard deviation σ and adding it to the mean vector μ:

$$z = g_\phi(\mathbf{x}, \epsilon) = \mu_x + \sigma_\epsilon \odot \epsilon \tag{2.13}$$

where $g(.)$ is a reconstruction function and $\odot$ is element-wise product.

2.1.4 Generative Adversarial Networks (GAN)

Similar to VAE, Generative Adversarial Networks (GAN) are capable of generating new data by sampling from the latent space and decoding it back into the original dimensional space. Their architecture consists of two elements: a generator, which produces outputs similar elements to the trained dataset, and a discriminator that aims to discriminate between real and fake data. The two elements play a zero-sum game and they work against each other. Unlike VAEs, GANs do not aim to maximize log-likelihood directly [Du et al. (2022)].

The GAN objective can be described as:

$$\min_G \max_D \mathcal{L}(D, G) = \mathbb{E}_{\mathbf{x} \sim p(z)}[\log(1 - D(G(z))] \tag{2.14}$$

The goal of the discriminator part of the network is to learn hidden patterns in the data so that a real data point can be discriminated from the synthesized data by the generator. On the other side, the generator learns to generate data that is identical to the qualities of the original data, which in turn tricks the discriminator to accept the generated data as genuine. It should be noted that the generator never sees a real data point, but is trained by the signal back-propagated from the discriminator. When this scenario is brought to the molecule generation domain, the generator learns to generate molecules out of Gaussian noise. On the other hand, the discriminator is introduced to a batch of molecules mixed with real and synthetic data, and its loss is calculated based on whether the discriminator is capable of distinguishing the real data from the generated ones.

The failure of the discriminator will be counted as the success of the generator, hence its loss is calculated depending on how indistinguishable the data it is capable to generate. In the end, the discriminator and the generator will reach the Nash equilibrium [Bian and Xie (2021). However, training can be unstable because of the almost simultaneous optimization in case the gradient of one loss is being favored over the other. This may lead to a discriminator being more competent than the generator or vice versa.

2.2 Molecular Representations

Molecular representation is known to affect the extent to which the biochemical relationships can be captured by the model [Du et al. (2022)]. Although variations exist, two of the most popular representations are: the Simplified Molecular Input Line Entry System (SMILES) [Weininger (1988)] representation and 2-D graph representation of molecules in which atoms constitute vertices and bonds correspond to the edges between the vertices. Figure 2.3 illustrates a 2-D graph representation of a molecule along with its SMILES string representation. SMILES is a 1-D string representation of molecules and provides a relatively understandable representation of the molecule depending on its length. On the basis of which atom the string starts with, multiple SMILES strings can indicate the same molecule and cause ambiguity. In order to standardize a representation for a given molecule, canonicalization is proposed, which is provided by the RDKit library [Landrum (2016)]. The string is usually transformed into one hot encoding of each unique SMILES string symbol so that models like RNN can operate directly on the representation [Bian and Xie (2021)]. A typical flow of a SMILES representation-based generation process can be seen in Figure 2.2. There are two alternatives to represent molecules as strings, either by displaying the hydrogen atoms explicitly or implicitly.

The common belief in the drug generative modeling field is that graph representation enables the generative model to capture more details about the molecules,

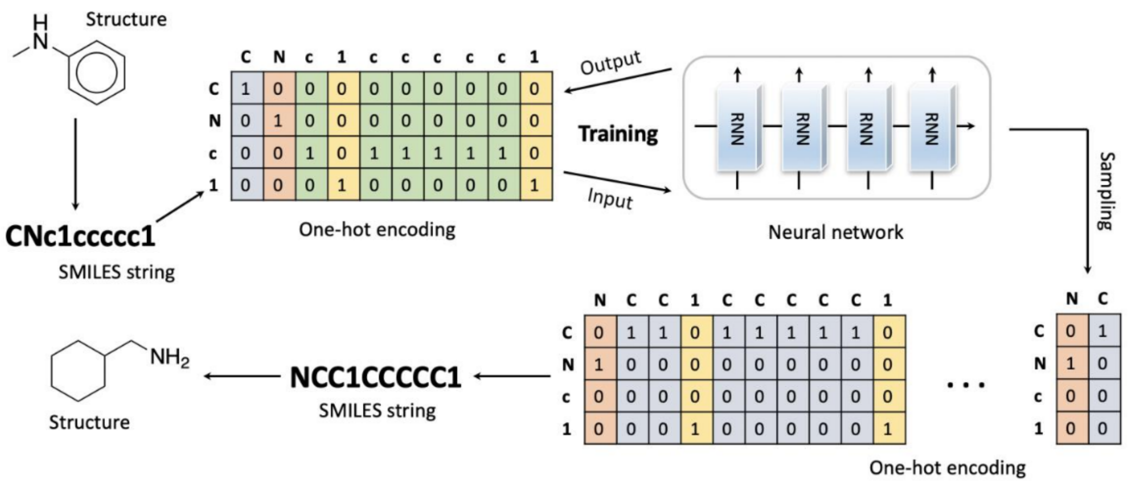


Figure 2.2: A representative flow of SMILES string-based generative process (taken from [Bian and Xie (2021)]).

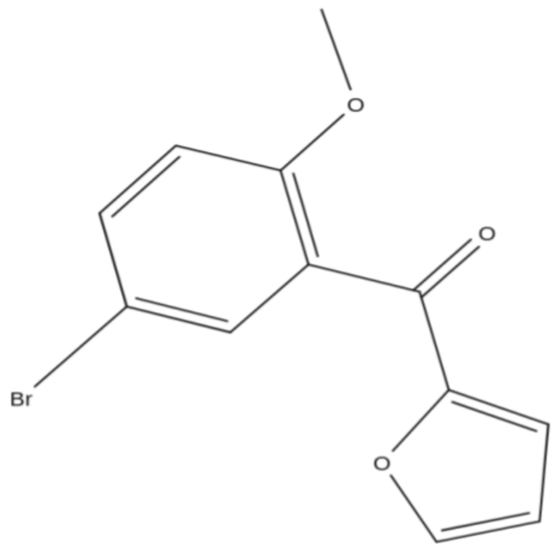


Figure 2.3: 2-D (top) and SMILES (bottom) representations of a molecule from the MOSES dataset.

since molecules are graph-alike structures by nature [Du et al. (2022)]. Although the molecular conformations are best represented with 3-D graphs, such a geometrical representation that respects the 3-D symmetry is not an easy task. Hence, 2-D representation is preferred by most of the state-of-the-art drug generative models. In graph representation, atom information is captured by a feature matrix, while bond information is expressed by an adjacency matrix.

2.3 Interpretability of Generative Models

Available interpretability methods for drug generative models are scarce, as opposed to the classification problem, where there are multiple off-the-shelf libraries readily provided [Kokhlikyan et al. (2020)]. In the context of generative modeling, the idea of interpretability mostly revolves around the concept of disentanglement [Singh and Ogunfunmi (2022b)][Eastwood and Williams (2018)], where each ground truth generative factor is mapped to at most one factor of variation in the latent space. However, most of the already existing methods for generative models do not translate well into the molecular generation domain, since the underlying ground truth latent factors are unknown for molecules. In the drug generative model domain, most of the work resembles case studies [Du et al. (2022)].

A couple of architecture improvements have been proposed [Higgins et al. (2017)][Chen et al. (2019)] for the VAE-based generative paradigm, encouraging the model to learn the latent space in a way that latent variables are independent of each other. Although isotropic Gaussian distribution encourages such independence, by itself is not enough to satisfy such a requirement. On the other hand, VAE-based drug generative models can be prone to suffer from disentanglement, due to the loss function it aims to optimize.

2.4 Hierarchical Generation of Molecular Graphs Using Structural Motifs

HierVAE Jin et al. (2020) is a VAE-based molecule generative architecture that aims to improve upon the limitations of earlier architectures that consider smaller building blocks such as atoms, which limit their applicability to smaller molecules, due to the requirement of assembling the neighborhood of substructures in one step. In contrast, HierVAE utilizes larger molecular motifs, thereby enabling it to work with more complex molecules. The novel approach enables the encoder to maintain a multi-resolution representation of molecules from fine to coarse fashion that ranges from atoms to connected motifs. The decoder operates in a symmetric way, processing the information from a coarse-to-fine fashion by joining one motif at a time, along with the decision of selecting which motif to merge with the emerging molecule from which bonding point.

The building block motifs are extracted from the molecule dataset depending on their frequency of occurrence. A motif belonging to a polymer is displayed in Figure 2.4. During the generation phase, molecules will be assembled by attaching motifs to the emerging molecule block. In turn, the decoder has to make a couple of sequential predictions: selection of the motif to be appended, which part of it to be attached, and which atom should be the point of contact with the partially generated emergent molecule. The mirroring encoder layer provides information about each consecutive decision made by the decoder. Iterative graph convolution enables fine-to-coarse encoding which relies on the information provided by the previous layer.

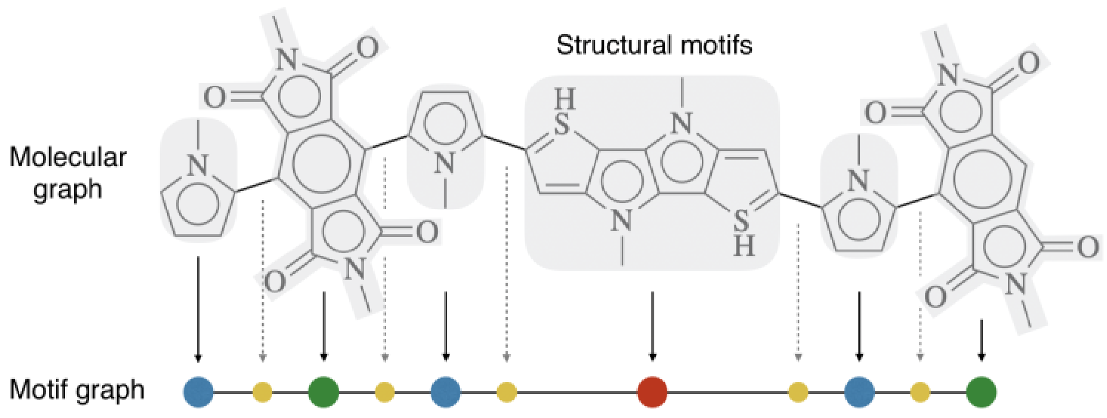


Figure 2.4: Motifs with corresponding color projections of parts highlighted in grey (taken from [Jin et al. (2020)])

Following Jin et al. (2020), motifs from the given dataset D are expressed as $\mathcal{S}_i = (\mathcal{V}_i, \mathcal{E}_i)$ as a subgraph of molecule $\mathcal{G}$ consists of atoms $\mathcal{V}_i$ and bonds $\mathcal{E}_i$. For each molecule, motifs $\mathcal{S}_1 \dots \mathcal{S}_n$ are extracted in a way that their union will result in the whole graph, that is $\bigcup_i \mathcal{V}_i$ and $\bigcup_i \mathcal{E}_i$. In order to carry out the extraction, molecule $\mathcal{G}$ will be partitioned into disconnected parts by removing the bridge bonds without breaking the chemical validity constraint. After exposing all the molecules in the dataset to this procedure, a molecule vocabulary V_S is obtained.

The autoencoder approach is designated to model the probability of graph $\mathcal{G}$ to be a joint distribution of motifs $\mathcal{S}_1 \dots \mathcal{S}_n$ along with attachments $\mathcal{A}_1 \dots \mathcal{A}_n$. Such attachments, expressed as $\mathcal{A}_i = \{v_j \mid v_j \in \bigcup_k \mathcal{S}_i \cap \mathcal{S}_k\}$ are the intersection atoms between motif $\mathcal{S}_i$ and its neighbor motif. The auto-regressive factorization of $P(\mathcal{G})$:

$$P\mathcal{G} = \int_z P(z)\Pi_k P(\mathcal{S}_k, \mathcal{A}_k \mid \mathcal{S}_{<k}, \mathcal{A}_{<k}, z)dz \tag{2.15}$$

During each step of the generation process, the decoder appends a new motif prediction $\mathcal{S}_k$ along with the attachment prediction $\mathcal{A}_k$ before deciding how the new motif will be attached to the current graph prediction.

In order to enable the decoder to realize such hierarchical processing, the encoder is needed to supply the essential information in each decoding step. Therefore, a molecule $\mathcal{G}$ is represented by a hierarchical graph $\mathcal{H}_\mathcal{G}$ consisting of three main layers.

The motif layer provides the broadest representation with rough topological information about how the motifs are connected with each other. More precisely, the layer contains n nodes as motifs $\mathcal{S}_1 \dots \mathcal{S}_n$ together with m edges $\{(\mathcal{S}_i, \mathcal{S}_j) \mid \mathcal{S}_i \cap \mathcal{S}_j \neq 0\}$ for all of the intersecting motifs $\mathcal{S}_i, \mathcal{S}_j$. This tree-structured layer supplies information about motif prediction in the decoding step.

The attachment layer is designated to represent information about motif connectivity in finer detail. Node $\mathcal{A}_i = (\mathcal{S}_i, \{v_j\}$ contains the information of a specific attachment scenario of motif $\mathcal{S}_i$, where $\{v_j\}$ denote the atoms located in the intersection between motif $\mathcal{S}_i$ and one of its particular neighbor motifs. For every possible attachment configuration of a motif $\mathcal{S}_i$, a vocabulary of attachments $V_\mathcal{A}(\mathcal{S}_i)$ is constructed from the training subset. The task of attachment prediction during the decoding process is facilitated by this layer.

At the finer side of the representation spectrum, molecular graph $\mathcal{G}$ provides information on atom-level connectivities. Every atom node v is associated with a label a_v which keeps the atom type and charge information. Since there are multiple types of bonds possible between given two atoms, the edge (u, v) is also labeled with the bond type information b_{uv}. This layer's information is passed into the decoder for supporting the task of graph prediction.

In order to provide a connection between different levels of representation layers, certain edges are introduced between motifs and atoms. After obtaining the hierarchical graph $\mathcal{H}_\mathcal{G}$ with the addition of bridge edges, the graph is encoded by a hierarchical Message Passing Network (MPN). In their experiments, Jin et al. (2020) detect that, on average, the size of motif vocabulary $\mid V_\mathcal{S} \mid < 500$ and the attachment vocabulary size $\mid V_{\mathcal{A}}(\mathcal{S}_i) \mid \leq 10$, which are one hot encoded in order to be processed.

For each of the representational layers of the encoder, three MPNs are maintained: the encoding by MPN is expressed as $MPN_{\psi}(.)$ which is parameterized by ψ and a regular MLP as $MLP(x, y)$.

The encoding protocol starts from the atom level: atom layer of $\mathcal{H}_G$, represented as $\mathcal{H}_G^g$ after the encoding. Atom and bond embeddings of $\{\mathbf{e}(a_u), \mathbf{e}(b_{uv})\}$ are fed into the MPN. For T iterations, the MPN network propagates the message vectors between neighbor atoms, where at the end the network outputs the new representation $\mathbf{h}_v$ for each atom v:

$$c_G^g = \{h_v\} = MPN_{\psi_1}(\mathcal{H}_G^g, \{\mathbf{e}(a_u), \mathbf{e}(b_{uv})\}) \tag{2.16}$$

The second layer in the hierarchical representation, the attachment layer $\mathcal{H}_G^a$, takes $\mathcal{A}_i$ as input which is the concatenation of the embedding $e(\mathcal{A}_i)$ and summation of its atom vectors $\{\mathbf{h}_v \mid v \in \mathcal{S}_i\}$:

$$\mathbf{f}_{\mathcal{A}_i = MLP\left(e(\mathcal{A}_i), \sum_{v \in \mathcal{S}_i} \mathbf{h}_v\right)} \tag{2.17}$$

Input features of each edge $(\mathcal{A}_i, \mathcal{A}_j)$ is the embedding vectors $e(d_{ij})$ indicating the relative ordering between nodes $\mathcal{A}_i$ and $\mathcal{A}_j$ during the decoding step. Representations of the motifs are obtained by:

$$c_{\mathcal{G}}^a = \{h_{\mathcal{A}_i}\} = MPN_{\psi_2}\left(\mathcal{H}_G^a, \{f_{\mathcal{A}_i}\}, \{e(d_{ij})\}\right) \tag{2.18}$$

And for the last representational layer of the encoder, node $\mathcal{S}_i$ in the motif layer takes as input feature the concatenation of embedding $e(\mathcal{S}_i)$ and node vector $h_{\mathcal{A}_i}$ from the layer before:

$$\mathbf{f}_{\mathcal{S}_i} = MLP(e(\mathcal{S}_i), h_{\mathcal{A}_i}) \tag{2.19}$$

The motif representation is then obtained by:

$$c_{\mathcal{S}}^g = \{h_{\mathcal{S}_i}\} = MPN_{\psi_3}\left(\mathcal{H}_G^{\mathcal{S}}, \{\mathbf{f}_{\mathcal{S}_i}\}, \{e(d_{ij})\}\right) \tag{2.20}$$

As a result, a molecule $\mathcal{G}$ will be $z_{\mathcal{G}}$ represented as in the latent space, which is sampled by reparameterization trick with mean $\mu(h_{\mathcal{S}_1})$ and log variance $\Sigma(h_{\mathcal{S}_1})$ so that gradients can flow:

$$z_{\mathcal{G}} = \mu(h_{\mathcal{S}_1}) + exp(\Sigma(h_{\mathcal{S}_1})) \cdot \epsilon; \epsilon \sim \mathcal{N}(0, I) \tag{2.21}$$

The motif $\mathcal{S}_1$ is the first motif to be generated during the reconstruction process, identified as the root motif.

On the other hand, the decoder almost operates symmetrically to the encoder counterpart, where the molecule $\mathcal{G}$ is generated by expanding its hierarchical graph. During the decoding process, the model keeps a stack of frontier nodes $\mathcal{F}$ where $\mathcal{S}_k \in \mathcal{F}$ are the motif nodes that still have pending neighbors to be generated. The motifs are extended in a depth-first manner. For a node $\mathcal{S}_k$ at the top of the stack $\mathcal{F}$ at step k, the decoder predicts the following three, which are conditioned on latent vector $z_\mathcal{G}$:

$$\mathbf{p}_{\mathcal{S}_t} = softmax(MLP(\mathbf{h}_{\mathcal{S}_k}, z_\mathcal{G})) \tag{2.22}$$

equation 2.22 is the probability of motif $\mathcal{S}_t$ being attached to the motif $\mathcal{S}_k$. After deciding which motif to attach, the decoder has to predict which attachment configuration $\mathcal{A}_k$ of $\mathcal{S}_k$ should be the case, which can be considered as a classification problem over the vocabulary of attachments $V_\mathcal{A}(\mathcal{S}_t)$:

$$\mathbf{p}_{\mathcal{A}_t} = softmax(MLP(\mathbf{h}_{\mathcal{A}_k}, z_\mathcal{G})) \tag{2.23}$$

After predicting which atoms $v_j \in \mathcal{S}_t$ belongs to its intersection with the neighbor motif, the decoder makes a final prediction of how motifs $\mathcal{S}_t$ and $\mathcal{S}_k$ should be connected. This attachment is expressed as atom pairs $\mathcal{M}_{tk} = \{(u_j, v_j \mid u_j \in \mathcal{A}_k, v_j \in \mathcal{A}_t\}$ in which atoms u_j and v_j is bonded. Possible attachments M are formalized based on the atom vectors $\mathbf{h}_{u_j}$ and $\mathbf{h}_{v_j}$:

$$\mathbf{p}_M = softmax(\mathbf{h}_M \cdot z_\mathcal{G}) \tag{2.24}$$

$$\mathbf{h}_M = \Sigma_j MLP(\mathbf{h}_{u_j}, \mathbf{h}_{v_j}) \tag{2.25}$$

These sequential predictions made by the decoder cause results to be highly dependent on each consecutive step since the attachment prediction outlines the motif to be predicted. The motif and layer relationship is highlighted in Figure 2.5

The training objective of the model is the minimization of the negative ELBO, where teacher forcing is utilized in the generative steps and the order of generation based on the depth-first traversal over the ground truth molecule:

$$-\mathbb{E}_{z \sim Q}[logP(\mathcal{G} \mid z)] + \lambda_{KL}\mathcal{D}_{KL}[Q(z \mid \mathcal{G}) \| P(z)] \tag{2.26}$$

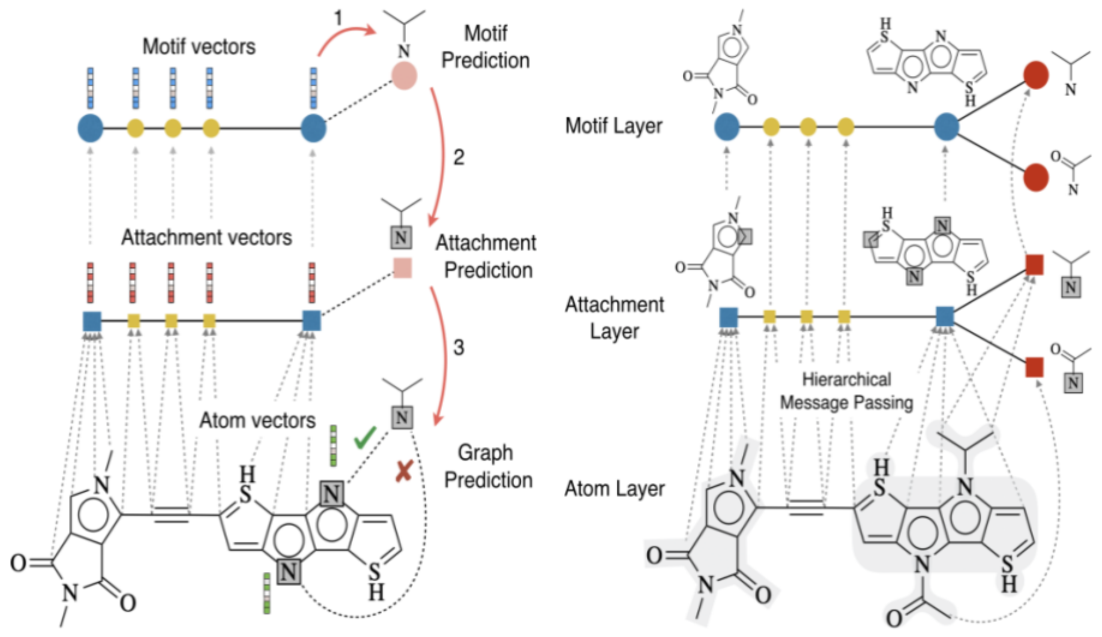


Figure 2.5: The motif and layer relationship highlighted for encoder (left) and decoder (right) (taken from [Jin et al. (2020)])

2.5 MoFlow: An Invertible Flow Model for Generating Molecular Graphs

MoFlow [Zang and Wang (2020)] is a unique approach that guarantees validity in the generated molecules and can re-construct the training data with 100% success, hence it is advantageous over the more conventional generative models. There have been a few Flow based generative models proposed for molecule generation[**?**][Madhawa et al. (2019)][Shi et al. (2020)], but MoFlow is the only architecture that can guarantee the validity of the molecules.

Overall, the invertible flow-based models' objective is to learn invertible mappings between the prior distribution and a more complex distribution through neural networks, which is also capable of learning the exact likelihood in a tractable manner, whereas the VAE-based models approximate the likelihood. To formalize, the flow model objective is to learn a sequence of invertible transformations $f_\Theta = f_L \circ \ldots \circ f_1$ between high dimensional complex data $X \sim P_\mathcal{X}(X)$ and same dimensional but simple enough to model $Z \sim P_\mathcal{Z}(Z)$ where independence in the latent space is assumed. In order to express the complex data in terms of the simpler distribution:

$$P_\mathcal{X}(X) = P_\mathcal{Z}(Z) \mid \det(\frac{\partial Z}{\partial X}) \mid, \tag{2.27}$$

where $Z = f_\Theta(X)$. In order to sample $\tilde{X} \sim P_\mathcal{X}(X)$, $\tilde{Z} \sim P_\mathcal{Z}(Z)$ will be sampled instead and then transformed as $\tilde{X} = f_\Theta^{-1}(\tilde{Z})$ by the reverse mapping of f_Θ. In order to train with exact likelihood, let $Z = f_\Theta(X) = f_L \circ \ldots \circ f_1(X)$, $H_l = f_l(H_{l-1})$ where $f_l(l = 1, \ldots, L \in \mathbb{N}^+)$ are invertible mappings, $H_0 = X, H_L = Z$ where $P_\mathcal{Z}$ is isotropic standard Gaussian distribution with independent dimensions. In order to express the log-likelihood of X:

$$\begin{gathered} \log P_\mathcal{X}(X) = \log P_\mathcal{Z}(Z) + \log \mid \det(\frac{\partial Z}{\partial X}) \mid = \\ \sum_i \log P_{\mathcal{Z}_i}(Zi) + \sum_{l=1}^{L} \log \mid \det(\frac{\delta f_l}{\delta H_{l-1}}) \mid \end{gathered} \tag{2.28}$$

in which $P_{\mathcal{Z}_i}(Zi)$ denotes the probability of the i^{th} dimension of Z and $f_\Theta = f_L \circ \ldots \circ f_1$ is the invertible neural network that is aimed to be learned.

Similar to the HierVAE model, MoFlow also operates on the graph representation of the molecules, where atoms are represented by a feature matrix $A \in \mathbb{R}^{nxk}$ in a one-hot-encoded form and bonds are represented as the bond tensor $B \in \mathbb{R}^{cxnxn}$, where n, k, and c are the number of type of atoms, the maximum number of atoms and the number of bond types respectively. As a result, a molecule $M = (A, B)$ can be framed as an undirected graph with multiple types of edges and vertices. The objective can be roughly expressed as learning molecule generative model $P_\mathcal{M}(M)$ that captures the probability of sampling molecule M from the molecule graph distribution $P_\mathcal{M}$. In order to consider both the atom and the bond structures of the graphs, $P_\mathcal{M}(M)$ can be reformulated as :

$$P_\mathcal{M}(M) = P_\mathcal{M}(A, B) \approx P_{A|B}(A \mid B; \theta_{A|B}) P_B(B; \theta_B), \tag{2.29}$$

where P_B is the bond distribution, $P_{A|B}$ is the conditional distribution of atoms given bonds, θ_B and $\theta_{A|B}$ are the learnable parameters of the model. In contrast to GAN and VAE, the exact likelihood can be obtained by:

$$\underset{\theta_B,\ \theta_{A|B}}{\operatorname{argmax}} \mathbb{E}_M = (A, B)_{\sim P_{M-data}} [\log P_{A|B}(A \mid B; \theta_{A|B}) + \log P_B(B; \theta_B)] \tag{2.30}$$

The two building blocks of the model are the graph conditional flow that conditionally learns atom matrices given bond tensors and a flow that learns bond tensors.

The graph conditional flow defines two flows that transform A given B into a conditional variable in latent space $Z_{A|B} = f_{A|B(A|B)}$ which is an isotropic Gaussian

$P_{\mathcal{Z}_{A|B}}$. In order to get the conditional probability of atom features given bond $P_{A|B}$, a conditional variation of equation 2.27 is implemented.

On the other hand, in order to get bond tensors, a variant of Glow [Kingma and Dhariwal (2018)] is implemented. Glow is a simple 1×1 invertible convolution, which will be utilized to process graph information. The goal is to learn an invertible mapping where the transformed latent variable $Z_B = f_B(B)$ is isotropic Gaussian. The equation 2.28 can be applied to logarithmic probability of bonds $\log P_B(B)$ and bond tensors can be produced by reverse mapping $\tilde{B} = f_B^{-1}(\tilde{Z})$ in which $\tilde{Z} \sim P_{\mathcal{Z}}(Z)$.

In order to ensure validity, a valence constraint over each atom is defined as:

$$\sum_{c,j} c \times B(c,i,j) \leq Valency(Atom_i) + C_h \tag{2.31}$$

where C_h is the formal charge. Instead of the reject-sampling paradigm which is applied by many autoregressive models, each molecule M will be checked for its atoms whether all of the atoms obey the valence constraints or not. In case of success, the longest connected component will be returned and the validity check procedure will be terminated. In case of detecting a violation of valence constraint, bonds of the atom will be sorted by their order, and the smallest bond will be removed. The procedure will be implemented once more until no violation is detected. This way, the molecule will be minimally modified and the largest component preserved. The overall structure of the MoFlow architecture is represented in Figure 2.6.

2.6 ChemSpacE: Interpretable and Interactive Chemical Space Exploration

ChemSpacE [Du et al. (2023)] is an attempt to explore latent spaces constructed by molecule generative models in a model-agnostic way. The work aims to translate previous methods proposed for image generative models [Yang et al. (2020)] into the molecule generation domain. The approach makes certain assumptions for the latent space: structurally or functionally similar molecules are mapped in close proximity to each other in the latent space, and interpolation between two molecules in the latent space results in smooth changes in the structures and properties in the molecules. Therefore, by identifying the separation hyperplanes that drive certain property changes in the latent space, the method claims to be capable

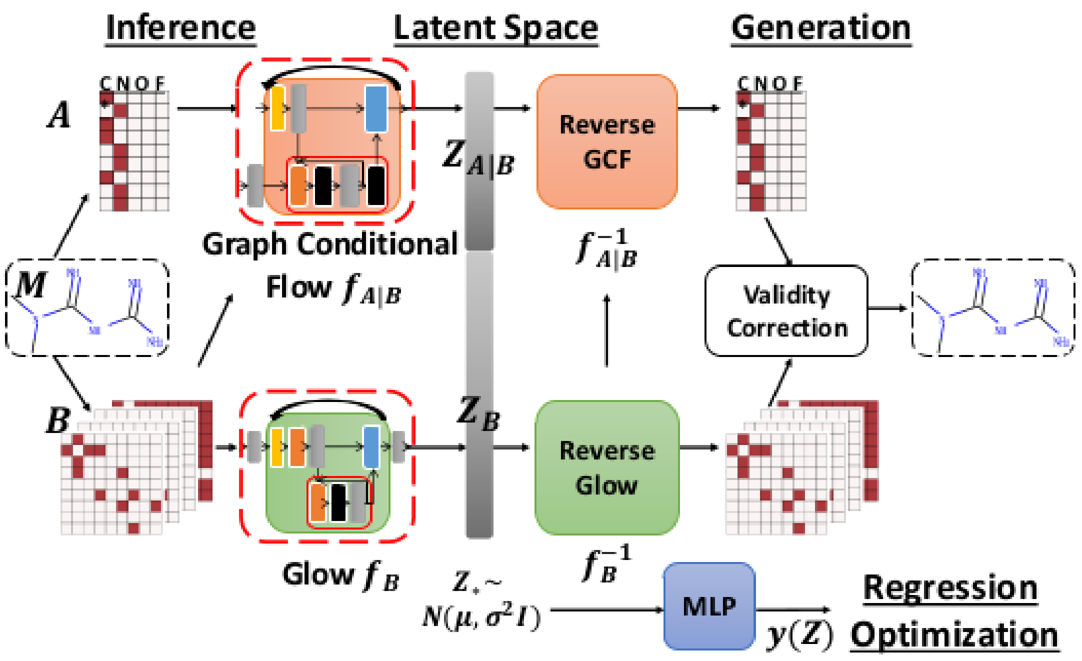


Figure 2.6: MoFlow architecture (taken from [Jin et al. (2020)]).

of navigating the latent space of any generative model and detecting the directions which lead to smooth property changes in the decoded molecules. Therefore, molecular properties can be optimized to a certain extent depending on the downstream task.

Also, to quantify the success of such manipulations, the work proposes two new metrics: Strict Success Rate (SSR) and Relaxed Success Rate (RSR). In the former, only the monotonic improvements in the selected properties are considered an absolute success, whereas, in the latter various threshold values are set to separate the success from the failure:

$$\phi_{SPC}(x,k,f) = 1[\forall i \in [k], s.t., f(x^{(i)}) - f(x^{(i+1)}) \leq 0] \tag{2.32}$$

$$\phi_{SSC}(x,k,\delta) = 1[\forall i \in [k], s.t., \delta(x^{(i+1)}, x^{(1)}) - \delta(x^{(i)}, x^{(1)}) \leq 0] \tag{2.33}$$

$$\phi_{DIV}(x,k) = 1[\exists i \in [k], s.t., x^{(i)} \neq x^{(1)}], \tag{2.34}$$

where f is a property function of molecule x, k is the number of steps when manipulating the molecule along the detected property boundary and δ is the structural similarity of a pair of molecules. Therefore, the SSR is defined as:

$$\frac{1}{|P| \times |X|} \sum_{p \in P, x \in X} 1[\phi_{SPC}(x_p, k, f_p) \wedge \phi_{SSC}(x_p, k) \wedge \phi_{DIV}(x_p, k)], \tag{2.35}$$

where P is the property space $f_p(X)$. The metric will discard any manipulation that is not monotonic, therefore the RSR can be more appropriate:

$$\phi_{SPC}(x,k,f,\epsilon) = 1[\forall i \in [k], s.t., f(x^{(i)}) - f(x^{(i+1)}) \leq \epsilon] \tag{2.36}$$

$$\phi_{SSC}(x,k,\delta,\gamma) = 1[\forall i \in [k], s.t., \delta(x^{(i+1)},x^{(1)}) - \delta(x^{(i)},x^{(1)}) \leq \gamma] \tag{2.37}$$

$$\phi_{DIV}(x,k) = 1[\exists i \in [k], s.t., x^{(i)} \neq x^{(1)}]. \tag{2.38}$$

By introducing ϵ and γ, a certain amount of deviation between consecutive manipulations will be tolerated by the metrics for property and structure similarities respectively. For the property similarity constraint, two variants are proposed; a local relaxed constraint and a local relaxed constraint:

$$\begin{gathered} RSR - L(P,X,k,\epsilon_l,\gamma) = \frac{1}{|P| \times |X|} \\ \sum_{p\in P, x\in X} 1[\phi_{RPC}(x_p,k,f_p,\epsilon_l) \wedge \phi_{RSC}(x_p,k,\gamma) \wedge \phi_{DIV}(x_p,k)], \end{gathered} \tag{2.39}$$

$$\begin{gathered} RSR - G(P,X,k,\epsilon_g,\gamma) = \frac{1}{|P| \times |X|} \\ \sum_{p\in P, x\in X} 1[\phi_{RPC}(x_p,k,f_p,\epsilon_g) \wedge \phi_{RSC}(x_p,k,\gamma) \wedge \phi_{DIV}(x_p,k)], \end{gathered} \tag{2.40}$$

When compared with the global constraint $RSR - G$, the local constraint $RSR - L$ emphasizes the model's capability to manipulate a specific property.

The task of finding property boundaries that separate each molecular property in the latent space is a very crucial task for the method to detect steerable directions in the latent space. After identifying boundaries, scaled surface normal vectors $n \in \mathbb{R}^l$ will be added to the latent vector of a molecule, causing a smooth change in the particular property value of the emergent molecule. The distance from a given sample z to the separation hyperplane is expressed as:

$$d(z,n) = n^T z \tag{2.41}$$

As latent vector z is manipulated in a way that it gets closer to the property boundary, the properties of the modified molecule change smoothly, as it is assumed that the property p and z are linearly dependent:

$$f_P(g(z)) = \alpha \cdot d(z,n) \tag{2.42}$$

As before, f_P is the oracle function of property P and α is the scaling factor that tunes down the surface normal vector of the property boundary. Generalizing the concept for multiple properties:

$$f_P(g(z)) = AN^T z \tag{2.43}$$

where matrix $A = Diag(a_1, \ldots, a_m)$ is the diagonal matrix of linear coefficients of each of the m molecular properties, and $N = [n_1, \ldots, n_m]$ consists of surface normal vectors of separation boundaries of each of m properties. Molecular properties P follow a multivariate normal distribution as:

$$\mu_P = \mathbf{E}(AN^T z) = AN^T \mathbf{E}(z) = 0 \tag{2.44}$$

$$\Sigma_P = \mathbf{E}(AN^T z z^T) N A^T = AN^T N A^T \tag{2.45}$$

If normal vectors in N are orthogonal to each other and Σ_P is a diagonal matrix, then molecular properties P are disentangled. In practice, not all of the properties are uncorrelated, and the entanglement is expressed as $n_i^T n_j$ where i and j denote molecular properties in P.

Lastly, the identified boundary and the direction that drives the property change can be used to manipulate molecule z to obtain a new molecule that has a higher(or lower) property value adjusted by the scaling parameter α:

$$z^{'} = z + \alpha n \tag{2.46}$$

Therefore, the resultant property value by this manipulation is:

$$f_P(g(z + \alpha n)) = f_P(g(z)) + k\alpha, \tag{2.47}$$

where k serves as a scaling factor to express the difference from the initial property value of latent code z. This configuration will work for single property manipulation cases. In contrast, when multiple properties correlate with each other, the amount of disentanglement between the two properties will signify the manipulation vector's magnitude. The emergent direction is nothing but the disentangled and positively correlated attributes of the directions:

$$n = n_1 + (1_{[n1 \odot n2 \geq 0]}) \odot n_2. \tag{2.48}$$

2.7 Dimensionality Reduction

T-Distributed Stochastic Neighbor Embedding (t-SNE) [Van der Maaten and Hinton (2008)] is a visualization method that projects the data into 2 or 3 dimensions. The technique is derived from the stochastic neighbor embedding method where optimization is easier than the earlier method and the quality of projections is improved by minimizing the tendency of clustering towards the center. It is assumed that the data lives in a lower dimensional manifold and t-SNE is able to capture the structure of such latent information. Hence, the method is capable to capture the local structure while still providing insights into the global structure.

Uniform Manifold Approximation and Projection (UMAP) [McInnes et al. (2018)] is another method mainly used for reducing high-dimensional data for multiple purposes, including visualization. Similar to t-SNE, the method also learns the manifolds to reduce provided data. It is inspired by Riemannian geometry and algebraic topology. Moreover, being a newer approach, it can scale better than t-SNE hence the computational cost is lower. Also, it is argued that the global structure of the original data is preserved in greater detail when compared against t-SNE.

2.8 Clustering Algorithms

HDBSCAN [Campello et al. (2013)] is a density-based clustering algorithm that extends DBSCAN [Ester et al. (1996)] by incorporating the concept of hierarchies into the clustering. The constructed hierarchy tree signifies the large clusters emerging from the data points. In order to get flat clusterings, where the tree is cut at a level that the clusters consist of only major clusters by taking into account the density information. To do so, the stability of the clusters is maximized.

On the other hand, K-Means [Hartigan and Wong (1979)] is one of the earliest clustering algorithms that survived the test of the time. The algorithm divides M points in N dimensions into K clusters, in a way that within cluster elements' sum of squares is minimized. The clustering will balance when no movement enables obtaining less sum of square values for any of the clusters. However, the method is able to detect only spherical-shaped clusterings, which are typically Gaussian distributions. Moreover, in cases where the density varies a lot from region to region, the method is known to fail to obtain high-quality clusterings. Although there are more modern approaches, the simplicity of the idea and its scalability still makes the algorithm popular as a baseline method.

Chapter 3

Materials and Methods

Among multiple options of the generative modeling frameworks for molecular generation, HierVAE [Jin et al. (2020)] and MoFlow [Zang and Wang (2020)] stand out due to the novel approaches they offer over the basic methods. To compare, a baseline method of a VAE from MOSES [Polykovskiy et al. (2020)] will also be of use as a comparison reference. All the models will be trained on the MOSES dataset since it is designated as a benchmark dataset to quantify the generative capabilities of drug generative models. For further details about the model architecture and parameters, please refer to Appendix A.

During the training of VAE-based architectures, the balance between the reconstruction loss and the KL loss components is of great importance, so that the model can both maintain a uniform latent space and reconstruct input molecules accurately. As proposed in [Higgins et al. (2017)], assigning a coefficient β to the KL term in the VAE loss further brings stability to the training. In practice, scheduling the weight penalty of the KL term during training leads to the best results. A popular approach is to linearly increase the penalty coefficient of β to a pre-determined maximum of $\max \beta$. At the first stages of the training, the model will be similar to a plain Autoencoder, since the KL term weight will be 0. As the training progresses into further epochs, the KL term starts to have some weight, therefore the encodings of the molecules resemble more and more distributions as opposed to being mere points so that overlap between encoded distributions can enable extrapolation. Although the ideal value may depend on the dataset, Yan et al. (2020) recommends $max\beta = 0.1$ for the ZINC250K[Irwin and Shoichet (2005)] subset. Since MOSES is a subset of the ZINC dataset, the same value can potentially serve as a good maximum. In order to compare the effect of $\max \beta$ with the linear annealing schedule, the VAE-based models will be trained with values of $\max \beta = 0.1$ and $\max \beta = 1$.

The assumptions made about the latent space of generative models, such as continuity and mapping of similar elements to the close neighborhood of each other are yet to be validated, since there are a couple of pitfalls known to challenge their validity. To test such assumptions, Latent encodings of selected sub-set of training molecules will be projected into the latent space. In order to do so, the high-dimensional latent space will be reduced with the dimensionality reduction algorithms t-SNE and UMAP and the corresponding encodings will be visualized. To test if there are meaningful clusterings in the latent space, the Physicochemical, Topological, Lipinski, and BCUT2D molecule descriptors will be calculated for the selected molecules. The descriptor features of each approach will be filtered based on the information content.

The extracted features will be used as input to the HDBSCAN clustering algorithm, where the clustering quality with the selected hyper-parameters will be judged by the silhouette coefficient of the resulting clusters.

After observing whether the basic assumptions on the latent space hold or not, the latent space exploration method ChemSpacE will be implemented for each model. Boundaries in the latent space will be discovered for a specified number of properties. For the experiments, generally accepted molecular properties of quantitative estimation of drug-likeness (QED), Wildman-Crippen LogP value (MolLogP), molecular weight (MolWt), a topological index meant to quantify molecular complexity (BertzCT), distance-based topological index (BalabanJ), connectivity topological indices (Chi1n), and the number of heavy atoms in the molecule (HeavyAtomCount) will be considered for boundary calculation. After identifying the boundaries, a number of molecules will be generated along with the direction of the surface normal vector of each boundary, and the resulting molecule properties will be inspected.

In order to quantify the obtained success of identifying boundaries and the resultant change in chemical properties of the extrapolated molecules, the SSR and RSR, defined in the previous chapter, will be calculated.

3.1 Model Training

3.1.1 Baseline VAE

The baseline VAE model is based on the string representation of molecules. For the training of the generative model, a subset of 200,000 molecules from the MOSES dataset is randomly selected. Later, the molecules are sorted in descending order in order to be fed to the RNN layer. After the addition of the beginning of a sentence (bos) and end of a sentence (eos) tokens, each string is padded to match the length of the longest string. Before feeding the SMILES strings to the model, each string character is mapped to an integer. Next, each integer vector is transformed into one-hot-encoded vectors. The baseline VAE model's encoder consists of a bi-directional GRU of hidden size 256 with a linear output layer of output size 128, which corresponds to the latent space dimensionality. The encoder outputs both mean and log-variance vectors for the molecules. The decoder consists of 3 layer GRU of 512 hidden dimensions with intermediate dropout layers with a dropout probability of 0.2. A batch size of 128 and gradient clipping with a value of 50 are set along with KL term weight linearly increasing from 0 to $\max \beta$ during training. Adam optimizer with a learning rate of 3.10^{-4} is preferred and the model is trained for 100 epochs. In order to observe the effect of the KL penalization coefficient, the model is trained with multiple configurations with $\max \beta$ values of 0.025, 0.1, and 1.0 respectively.

3.1.2 HierVAE

Prior to the forward pass, the SMILES strings of the molecules must be converted into a graph representation. Atoms and bonds are represented with atom and bond matrices respectively. The sizes of the matrices are dependent on the dataset at hand. In the case of the MOSES dataset, the longest molecule consists of 27 atoms and there are 7 types of atoms the molecules consists of. In order to specify which kind of a bond (edge) two connected atoms (vertices) have, a list of bond types is introduced with a length of 4, representing; a single bond, double bond, triple bond, and aromatic bond. Also, the integer 0 is appended to the list of atoms to express the non-existence of an atom in the feature(atom) matrix. Similarly, an extra channel is introduced to the bond(adjacency) matrix to indicate no bond between atoms. As a result, the feature matrix is of size 5x27 and the adjacency matrix is of size 5x27x27 obtained. The HierVAE model consists of a hierarchical message passing (HMPN) encoder and an HMPN decoder, which both utilize LSTMs. Each maintains a three-layer representation, motif level, intermediates level, and graph level where the latent space is of dimensionality 32. The model is trained for 20 epochs with a batch size of 50. An exponential learning rate scheduler with the annealing rate of 0.9 is applied where Adam optimizer with the learning rate of 10^{-3}

is preferred. Gradients are clipped at 5.0 to prevent exploding gradients. Similar to baseline VAE, $\max \beta$ values of 0.1 and 1.0 are selected. After exceeding the warm up phase of 10,000 steps, the weight coefficient increased linearly during the training phase.

3.1.3 MoFlow

Similar to the HierVAE model, MoFlow operates over the graph-based molecule representation paradigm. Hence every molecule in the dataset is processed into adjacency and feature matrices so that a multi-resolution representation of a molecule graph can be maintained by the model. The model is trained for 50 epochs with a batch size of 12 and a learning rate of 1×10^{-3}. The exponential learning rate scheduler of a decay rate of 0.9 is applied during the training, which takes effect every 450 steps during the training excluding the first 1000 steps as a warm up. In order to prevent overfitting, dropout with a rate of 0.2 is applied. Different from the VAE-based approach, the latent space is of the same dimensionality as the original data, which is $5 \times 27 \times 27 + 5 \times 7$ that corresponds to the addition of sizes of adjacency and feature tensors respectively.

3.2 Latent Space Clustering

In order to visualize the latent space, the space should be reduced to 2 or 3 dimensions. For that, UMAP and t-SNE algorithms are applied to the latent representations of randomly selected 15,000 molecules from the training data. The molecules are first subjected to the forward pass, then the outputs are fed into the reduction algorithm. For VAE-based models, the latent μ vectors of each molecule are selected since the re-parameterization trick of VAE-based models requires the latent representations to be broadcasted into z vectors to enable backpropagation, resulting in diluted representations by the addition of random normal noise.

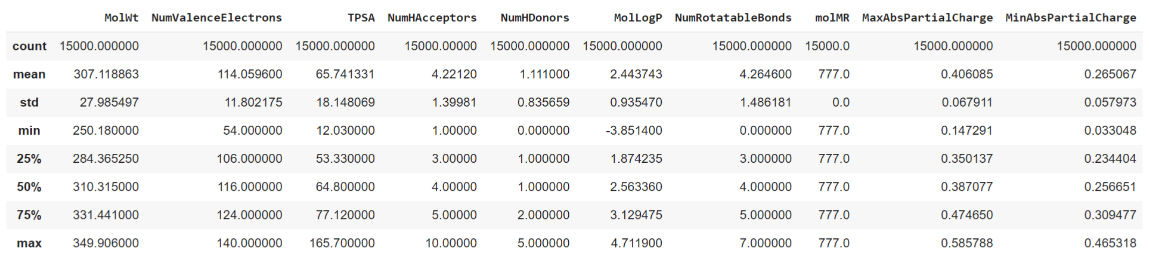
Following the suggestion in [Trozzi et al. (2021)], the data is first reduced before the clustering, since in high dimensions, the Euclidean distance and closeness concept warps due to the curse of dimensionality. Moreover, UMAP is preferred since it is shown in [Trozzi et al. (2021)] to be superior over t-SNE and PCA-based reduction approaches when the computational costs and the balance is considered, and the method is able to preserve the structural information well enough to model protein dynamics. To test this claim, the Physicochemical, Lipinski, Topological, and BCUT2D property descriptors will be clustered both in their original dimensionality and in the reduced space, then the results will be compared based on the obtained silhouette coefficients [Rousseeuw (1987)] since real ground truth cluster information is not available. The silhouette coefficient is calculated as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{3.1}$$

where $a(i)$ is the distance between i^{th} data point and all other data points in its assigned cluster, $b(i)$ is the average distance between all other points in the closest cluster, and $s(i)$ is the silhouette coefficient of point i. To get the overall picture, all of the silhouette coefficients of all data points will be averaged. The score values range between $[-1, 1]$, where the score of 1 indicates the point i is very well separated from the nearest cluster to its assigned cluster. Values close to 0 mean the point is in the decision boundary of two neighboring clusters, and values close to -1 indicate incorrect cluster assignment for a given point i

Moreover, the HDBSCAN algorithm is preferred due to factors such as: unknown ground truth cluster number, robustness against the density differences in the latent space, and indifference to cluster shapes, which the K-Means[Hartigan and Wong (1979)] algorithm is known to suffer from. However, the clustering procedure will also be carried out with the K-Means algorithm to provide a comparison baseline.

Before reduction, the distribution of molecule descriptor values is needed to be normalized, since the range of values differs highly per feature as shown in 3.1.



	MolWt	NumValenceElectrons	TPSA	NumHAcceptors	NumHDonors	MolLogP	NumRotatableBonds	molMR	MaxAbsPartialCharge	MinAbsPartialCharge
count	15000.000000	15000.000000	15000.000000	15000.00000	15000.000000	15000.000000	15000.000000	15000.0	15000.000000	15000.000000
mean	307.118863	114.059600	65.741331	4.22120	1.111000	2.443743	4.264600	777.0	0.406085	0.265067
std	27.985497	11.802175	18.148069	1.39981	0.835659	0.935470	1.486181	0.0	0.067911	0.057973
min	250.180000	54.000000	12.030000	1.00000	0.000000	-3.851400	0.000000	777.0	0.147291	0.033048
25%	284.365250	106.000000	53.330000	3.00000	1.000000	1.874235	3.000000	777.0	0.350137	0.234404
50%	310.315000	116.000000	64.800000	4.00000	1.000000	2.563360	4.000000	777.0	0.387077	0.256651
75%	331.441000	124.000000	77.120000	5.00000	2.000000	3.129475	5.000000	777.0	0.474650	0.309477
max	349.906000	140.000000	165.700000	10.00000	5.000000	4.711900	7.000000	777.0	0.585788	0.465318

Figure 3.1: Un-normalized physicochemical property values.

For example, the feature MolMR of Physicochemical descriptors can be seen in figure 3.1 to have a variance of 0, hence the feature will be dropped since it does not contain any useful information. The same approach is applied to the rest of the feature descriptors. To determine the best set of clustering parameters for the clustering algorithms, the grid search approach is applied. A range of hyperparameter values for the minimum cluster sizes and the minimum sample values are provided for the HDBSCAN, whereas a different range of k values for the K-Means clustering algorithm is experimented with, and the best parameters are selected based on the silhouette coefficient obtained by the given hyperparameters.

After extracting the cluster structure for the properties, data points close to the cluster centers will be chosen as seed molecules and they will be fed as priors to the decoder. Later, 100 molecules will be generated with the expectation of obtaining molecules being mapped to the same clusters as the prior, although some deviations can be acceptable due to the stochastic nature of the VAE model, that is, broadcasting latent vectors by adding random normal noise during the reparameterization trick. This way the assumptions about the uniformity of the latent space and close proximity mapping of similar molecules will be tested.

3.3 Property Boundaries and Extrapolation

In order to discover the directions that drive property changes in the latent space, the ChemSpacE method will be applied to the latent spaces of the models. First, 15,000 molecules will be generated by the generative models, and their corresponding molecular property values will be calculated by the RDKit [Landrum (2016)] library. Then a Support Vector Machine Classifier [Hearst et al. (1998)] with a linear kernel will be trained, where the problem is framed as a bi-classification of properties. In order to extract positive and negative samples, a hyperparameter of the split ratio is defined, which determines how many of the generated molecules with the highest property scores will serve as positive data, where the rest will be labeled as negative. The procedure is repeated for all of the chemical properties of interest. After the identification of such boundaries, their surface normal vectors will be serving as a direction that drives given property changes in the latent space. However, The method relies on the concepts of uniform organization of the latent space and mapping of similar items in close proximity, therefore its success will be highly dependent on to the degree that those assumptions hold for a given latent space. Later, previously generated molecules that are used to calculate boundaries will be manipulated in k steps with the direction of the identified property boundary surface normal vectors. The manipulation steps k, and the manipulation range are hyperparameters to be selected, since a very large manipulation range may result in sampling from a part of latent space that is too far away from the given molecules neighborhood. To find a good set of hyperparameters, the grid search approach is applied and the results are evaluated based on the obtained success rate metrics.

In order to quantify the results, the SSR and RSR with different relaxation coefficients are calculated, so that the disentanglement and the quality of detected property boundaries can be objectively quantified.

Chapter 4

Experimental Results and Discussion

4.1 Training Results

The baseline VAE model with $\max \beta = 0.025$ was able to obtain high validity and uniqueness scores, although the novelty of the generated molecules is the lowest obtained among all the alternatives. The other two configurations of $\max \beta$ values of 0.1 and 1.0 performed nearly the same for the evaluation metrics. However, the validity obtained appears to be approximately 6% less from the first configuration.

On the other hand, the HierVAE model was able to achieve 100% molecule validity in both of the $\max \beta$ configurations, while $\max \beta = 1.0$ got a slightly better novelty score. When compared with the baseline VAE configurations, at least a 12% increase in novelty scores is obtained. Although not as drastic of an improvement as in the novelty scores, roughly 2.5% to 8% increase has been observed over the baseline.

Interestingly, the MoFlow architecture was capable of getting perfect validity, uniqueness, and novelty scores, beating all other experimented model configurations. It was expected from the model to get 100% validity since complete validity is explicitly enforced by the model. Exact scores can be seen in Table 4.1.

Model	Validity	Uniqueness	Novelty
MOSES VAE $\max \beta{=}0.025$	0.9737	0.998	0.6955
MOSES VAE $\max \beta{=}0.1$	0.9216	0.9987	0.8606
MOSES VAE $\max \beta{=}1.0$	0.9169	0.9985	0.8535
HierVAE $\max \beta{=}0.1$	**1.0**	0.9988	0.9893
HierVAE $\max \beta{=}1.0$	**1.0**	0.9953	0.9999
MoFlow	**1.0**	**1.0**	**1.0**

Table 4.1: Training results.

Although the performance difference between the baseline model and the state-of-the-art variants is noticeable, a more complex dataset that consists of longer molecules could potentially lead to greater differences in the evaluation metrics. State-of-the-art architectures can scale relatively well due to novel design choices ingrained into their architecture, whereas the baseline VAE model would suffer as it operates on the basis of character-level generation.

4.1.1 Latent Space Clustering Results

As discussed in chapter 2.1.3, the KL term is assumed to be the main actor that distinguishes the VAE paradigm from a mere encoder of data points. The more the KL term of VAE is penalized, the more the data points will be expressed as standard normal Gaussian distribution. Hence, it is expected to obtain overlaps between individual data distributions and cover the latent space extensively.

After training the baseline VAE model with three different $\max \beta$ values, it can be seen that when the model highly penalizes the KL term, the obtained latent space spreads more evenly in contrast to the lower KL penalization case, where the model tends to construct the latent space into denser regions. In Figure 4.1, latent spaces of models penalized with $\max \beta = 0.1$ and $\max \beta = 1.0$ are visualized with the UMAP reduction method. To see the t-SNE equivalent, please refer to Appendix B. In order to emphasize the local and global structure of the original data separately, the number of neighbors values of 15, 50, and 200 for UMAP and perplexity values of 40, 100, and 200 with 1000 iterations are experimented with for t-SNE.

The selected sub-set of 15K molecules is clustered with three different approaches: Clustering in the original dimensionality with HDBSCAN and K-Means, and clustering the properties after reduction with UMAP. As discussed in chapter 3.2, the results are evaluated based on the obtained silhouette scores. The score values range from $[-1, 1]$, where a value close to 0 indicates not-so-clear-cut clustering and scores close to 1 indicate good separation between the clusters.

The HDBSCAN algorithm detected 5 clusters for physicochemical descriptors in the UMAP-reduced data. The rest of the property clustering results can be seen in Appendix C. A silhouette coefficient of 0.5507 is obtained by clustering parameters of minimum samples of 40 and minimum cluster size of 2. On the other hand, clustering in the original dimensionality with the HDBSCAN method did

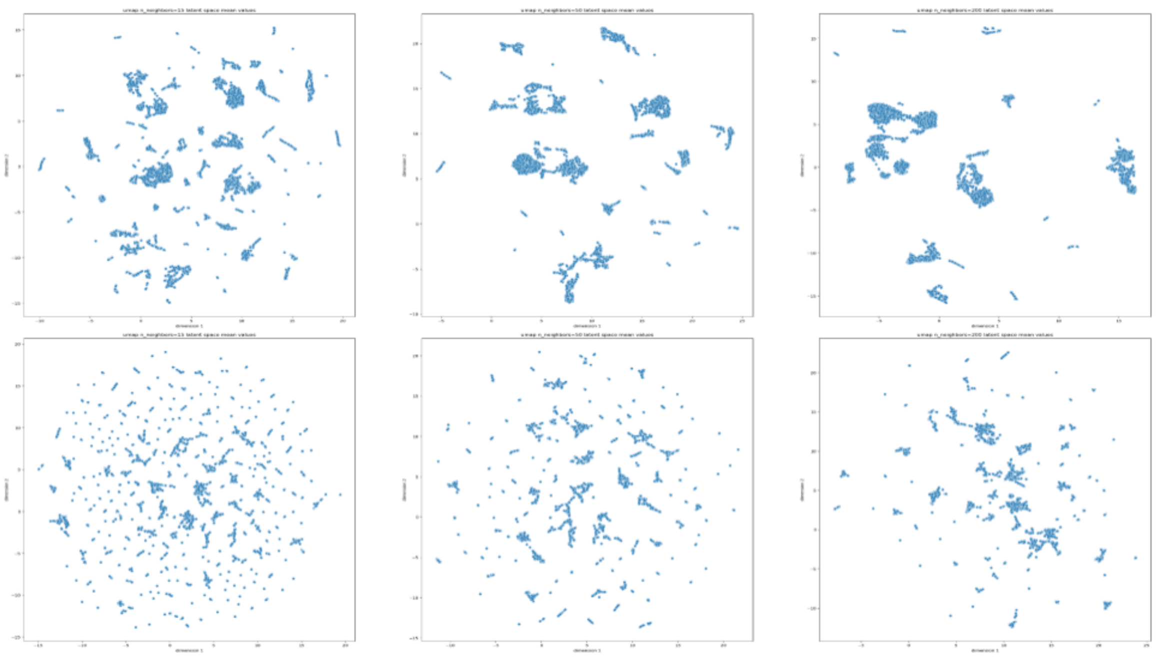


Figure 4.1: $\max \beta = 0.1$ (top) and $\max \beta = 1$ (bottom) latent space with UMAP.

not provide any stable clustering results, possibly due to the curse of dimensionality, where the size of the space increases at an extreme rate relative to the number of dimensions, hence causing data points to be equidistant to each other.

Similarly, the K-Means algorithm detected 7 clusters with a silhouette score of 0.13 as the highest among a range of $[6-30]$ for k values. As illustrated in Figure 4.2, it appears that the attempts of clustering in the original space have failed to extract plausible clusterings, whereas reduced space clustering is able to yield a higher clustering score for the molecular properties. However, it should be noted that the silhouette coefficient score might suffer from the bias of assigning higher scores for convex-shaped clusters, which density-based clustering methods are known to be inclined to produce. Since real cluster information is not available, there is no way of validating the real quality of the obtained clustering of molecules out of a single property descriptor. For the same reason, the clustering obtained by the reduced space can possibly be unrealistically well separated.

In order to validate the assumption of uniform organization in the latent space, the molecules situated at the center of each cluster are calculated, then they are used as a seed molecule to generate highly similar molecules, under the assumption that a generative model maintains similarity relations by mapping similar elements to the same neighborhood. If this assumption holds, then the mapping of close proximity is also expected to be observed. In order to detect the data point

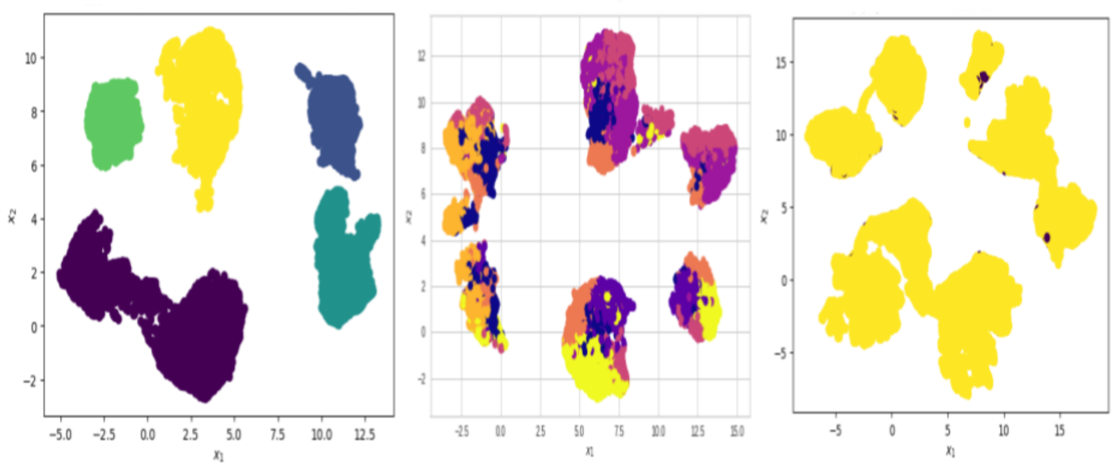


Figure 4.2: HDBSCAN clustering in reduced (left) and original (right) space and K-Means clustering (middle) in original space of physicochemical properties.

that is closest to the center for a given cluster, each data point's Euclidean distance from the cluster center is calculated, where the cluster center is the average of all the data points in the cluster. The data point with the minimum distance is selected as the closest point to the center.

However, the experiments suggest that the baseline VAE model does not appear to generate molecules that are mapped to the same cluster as the prior seed molecule. As displayed in Figure 4.3, only a fraction of the generated molecules fall into the same cluster as the seed molecule, whereas the rest of the molecules fall into all the remaining clusters. To further investigate whether the observed groupings of the molecules are due to the model's incapability to maintain the expected uniform organization or due to the obtained clustering, the generated molecules and the original prior molecule are projected onto the latent space in Figure 4.4. The similarity analysis based on Morgan Fingerprints of the generated and the seed molecule revealed that the mean dice similarity, a usual metric to compare molecule similarity, is approximately 24% in which the values range from 11% to 42%. This observation leads to the fact that the generated molecules are not only different in the considered molecular properties but also different in terms of molecular structure.

The projection displayed in Figure 4.4 can be considered as an indicator of pos-

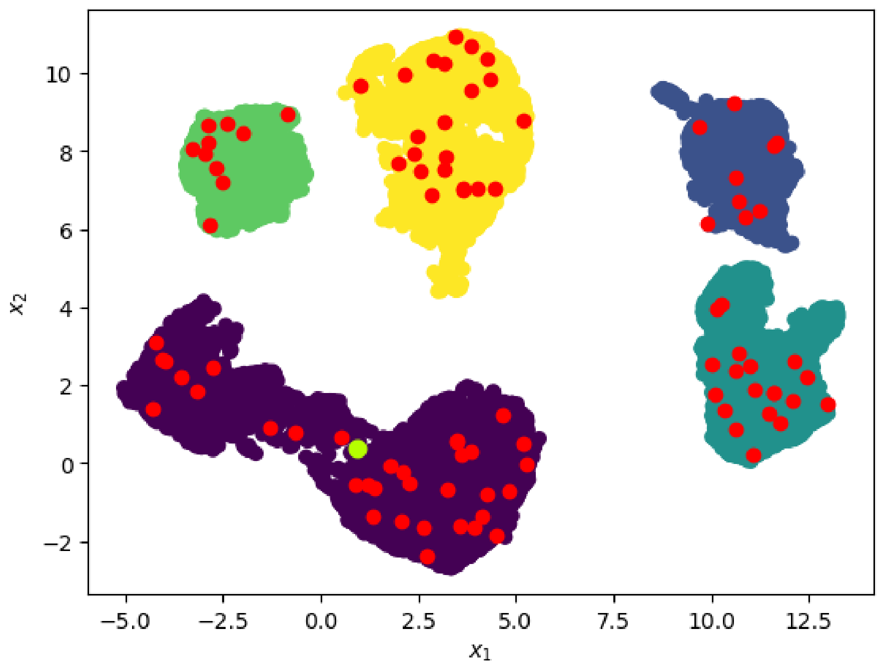


Figure 4.3: Projection of generated molecules (red) and the prior seed molecule (green) onto clusters.

terior collapse since the model appears not to focus similar molecules into a close neighborhood of each other. The VAE-based generative paradigm is known to suffer from such a phenomenon, where during the training the decoder tends to ignore certain sets of latent information, resulting in the learned variational distribution being close to the prior. Although it was previously believed that the issue occurs due to a strong decoder, [Lucas et al. (2019)] indicates that it is still possible for a model to still suffer under a not-so-powerful decoder. Moreover, [Singh and Ogunfunmi (2022a)] argues that the VAE models can suffer from inefficiencies when sampling from the high dimensional latent space since the prior is an isotropic Gaussian with L_2 Norm. In higher dimensions, the mass of the Gaussian distribution no longer concentrates around the mean, and instead of resembling a bell-shaped curve, it is similar to a uniform distribution on a hypersphere, where most of its mass is concentrated on the shell of the sphere.

4.1.2 ChemSpacE Exploration

To further investigate the latent space, the ChemSpacE approach was applied to all of the model variations, and the disentanglement aspect of each model's la-

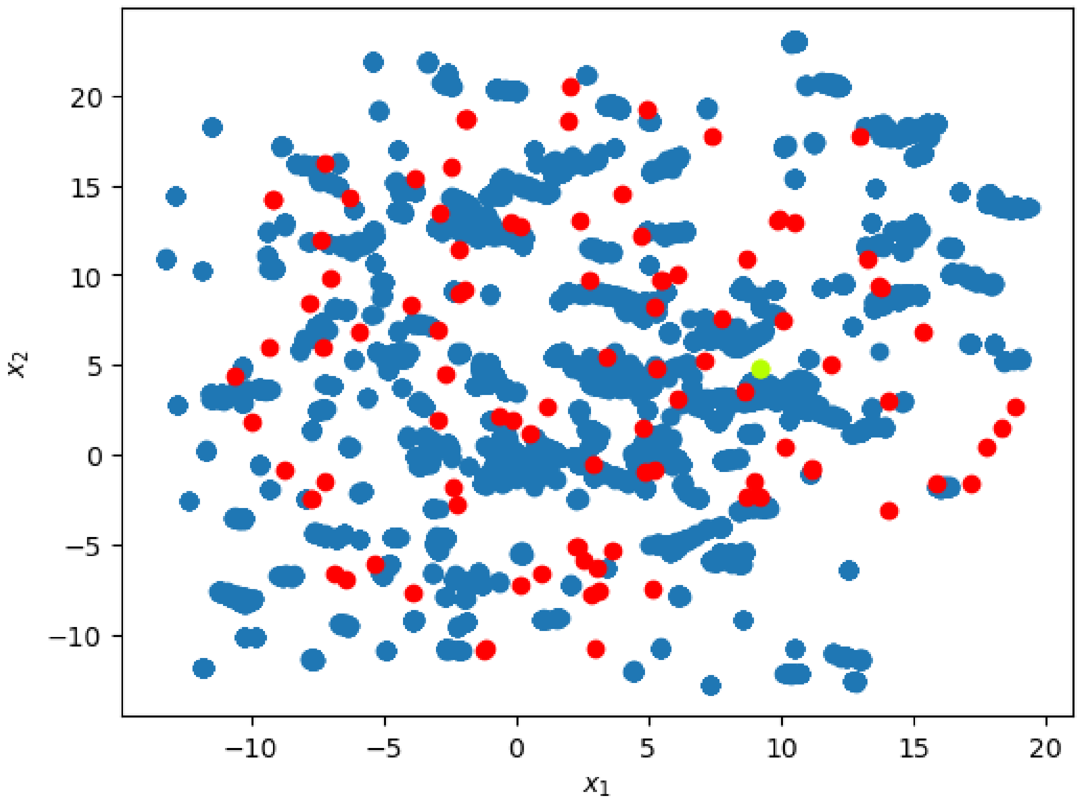


Figure 4.4: Projection of generated molecules (red) and the prior seed molecule (green) onto the latent space.

tent space is examined by the method. If the method is able to identify property boundaries and produce meaningful extrapolations, such as leading to a uniform increase or decrease in selected property values as the extrapolation goes, then it can be assumed that the model's latent space successfully achieves the disentanglement since in an entangled space such a concept would not be achievable. Therefore, it will be possible to navigate the synthesized space and manipulate molecules depending on the desired molecular properties.

In order to carry out the experiment, 200 molecules are sampled from each of the model configurations. After quantifying each of the generated molecule's molecular properties for 8 property descriptors via oracle calls from RDKit, the Support Vector classifier extracted the property boundaries out of the calculated property values by considering the task as a bi-classification problem. The grid search revealed that the results were the best on average with considering the obtained property values with a split of 10% positives and 90% negatives with training and validation splits of 70% and 30% respectively. In order to extrapolate, the obtained boundaries' surface normal vectors were scaled in a range of [-1, 1] in $k = 7$ steps and the resultant vector was added to the latent vector of each molecule, manipulating them gradually along the given boundary. Based on the resultant vectors, the SSR, RSR-Global (RSR-G) with 5% and 10% allowances are calculated as explained in equation 2.40. The global variant is preferred to demonstrate the edge cases of the allowed deviation in the monotonic increase & decrease trends. Figure 4.5 illustrates an example manipulation from the extrapolation step of the MoFlow model under strict manipulation and Figure 4.6 highlights structural differences of the molecules at the opposite sides of the manipulation spectrum.

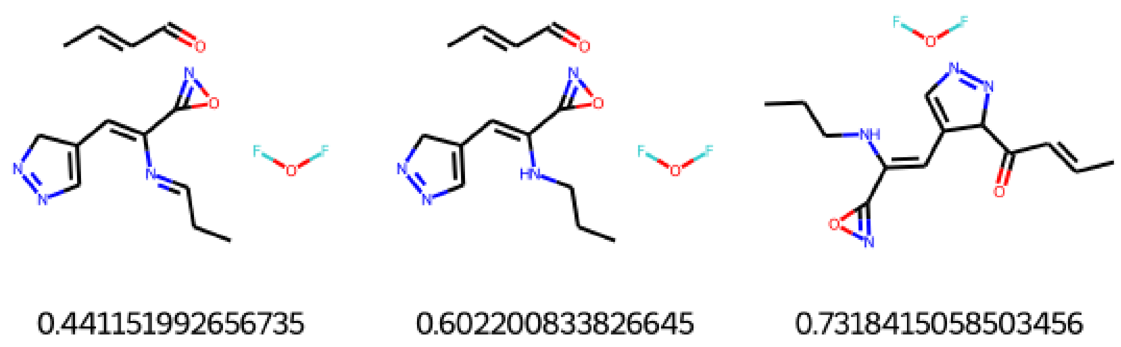


Figure 4.5: Molecule extrapolation along the QED property boundary with scores (bottom).

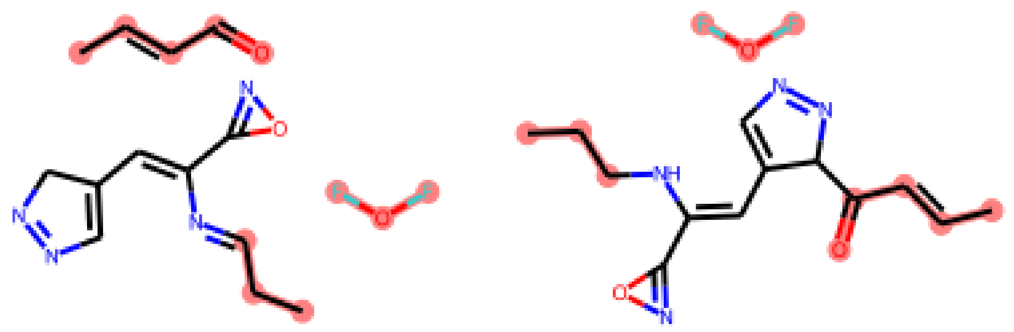


Figure 4.6: Highligted differences between the molecules at the opposite ends of the extrapolation spectrum.

Table 4.2 provides the achieved scores of SSR, RSR(0.05), and RSR(0.10) obtained by each model for the average of 8 property extrapolations. In order to see per property descriptor scores individually, please refer to Appendix D.

Model	Strict Success Rate(SSR)	$RelaxedSuccessRate - Global(RSR-G)_{\delta=\epsilon=0.05}$	$RelaxedSuccessRate - Global(RSR-G)_{\delta=\epsilon=0.1}$
MOSES VAE $_{\max\beta=0.1}$	0.000625	0.012125	0.34525
MOSES VAE $_{\max\beta=1.0}$	0.000375	0.01325	0.3836
HierVAE $_{\max\beta=0.1}$	0.2255	0.665125	**0.834**
HierVAE $_{\max\beta=1.0}$	0.19525	0.618125	0.8125
MoFlow	**0.665**	**0.730625**	0.7518

Table 4.2: ChemSpacE results

The results further confirm that the baseline VAE model suffers from posterior collapse [Singh and Ogunfunmi (2022b)], rendering the model to be unable to detect valid property boundaries and extrapolate molecules along them. On average, both of the baseline VAE configurations failed to produce molecules with monotonically decreasing property values in a strict manner. After introducing a 10% allowance to the monotonicity trend, the baseline models were able to produce some valid extrapolations. However, the relaxation amount of $\%10$, particularly under the global variant, could be too much of a relaxation to conclude the results as a definite success. On the other hand, the state-of-the-art VAE architecture HierVAE was able to achieve a 22% strict success rate, drastically improving upon the baseline model, especially when it is considered that SSR ignores even the slightest deviation in property values from the trend of monotonicity during the extrapolation. In addition to the model complexity being higher, the improved success can be attributed to the fact that the HierVAE architecture utilizes graph representation to process graph information, whereas the baseline VAE model is operating on the string representation of molecules, which is known to be less capable of capturing chemical details about molecules over the graph based counterpart and more prone to generate invalid molecules. Unsurprisingly, it is observed that during the extrapolation phase, baseline VAE often failed to obey atom-valence constraints and ring closures, causing the generated smiles string to be invalid molecules. When the condition is relaxed by as small as 5% of the global extrapolation range, an enormous jump of approximately 40% is achieved in success scores during extrapolation with HierVAE. The 5% threshold is a plausible compromise without giving up too much from the monotonicity trend.

Moreover, the tendency of receiving less success rate score by highly penalized $\max\beta$ configurations of VAE-based models appears to further validate the observations that the KL term is hindering models' ability to construct a latent space with the concept of close proximity mapping of similar molecules. The trade-off between obtaining meaningful encodings by emphasizing the reconstruction loss more against making encodings close to the desired distribution by penalizing the KL divergence term more appears to be one of the determining factors on whether the obtained space satisfies the expected uniform organization condition or not. The balance between two aspects of the training is of critical importance and it can be difficult to obtain.

More interestingly, the MoFlow model was able to achieve the highest SSR score with 66.5%, beating all other model architectures by a significant margin. The capability of the MoFlow to guarantee valid molecule generation appears to translate well into the molecule extrapolation procedure, in contrast to the inconsistent baseline VAE model. In terms of the steering capabilities in the latent space, MoFlow can navigate better in its synthesized space, despite the fact that its latent space is magnitudes larger than the rest of the experimented models.

On the other hand, there are chemical domain-specific pitfalls that limit the amount of success attainable for model interpretability, where most of the explainability approaches make oversimplifying assumptions about the latent space. The phenomenon called activity cliff [Stumpfe et al. (2019)] limits the attainable success with disentanglement-based approaches since molecules with highly similar structures may have disproportionately different physical properties, such as molecules that are active against the same target but having very different potencies. The relationship between the physical structure of a molecule and corresponding chemical property values is a highly complex and non-linear relationship, which is known as Structure-Activity Relationship (SAR) [Muratov et al. (2020)]. Moreover, despite the fact that graph-based representation can capture more details about molecular similarity over other alternatives, generative modeling cannot capture all of the complexity inherent to the molecular domain in an exhaustive manner yet.

It should be pointed out that the resulting scores of successful navigation could potentially be lower if a more complex dataset than MOSES is preferred, since MOSES is not the most complex dataset available for molecule generative tasks. Unfortunately, the number of publicly available datasets is also another aspect that hinders the efforts on drug generative model explainability, limiting most of the explainability attempts to a case-based scope. In general, it can be said that the more the dataset complexity increases by involving molecules consisting of more atoms, the more challenging it gets for the message-passing neural networks to propagate neighborhood information in a reasonable amount of iteration, hence giving way to diminishing gradients. On the other hand, SMILES representation-based RNN model architectures mostly fail to model such complex molecules. As an example, the longest molecule in the selected MOSES subset for training consists of 27 atoms, whereas polymers, molecules that consist of a larger number of atoms, can typically contain more than 70 atoms. Therefore, the applicability of certain model architectures is ultimately limited to the dataset at hand.

For the downstream task of learning data distribution, maintaining a disentangled latent space does not appear to be an absolute requirement: Although the baseline VAE model is not disentangled, it is still capable of learning the distribution of the training dataset well and generating valid, unique, and novel molecules, indicating the case that model interpretability and performance do not always go hand in hand or one is indicative of the other.

Chapter 5

Conclusions and Future Work

De novo drug design is one of the recent most successful applications of DL models. Such applications aspire to discover the vast latent space of chemically plausible molecules, which is not feasible to cover with traditional *in vitro* methods.

Among multiple paradigms proposed for drug generative models, the VAE and Reversible Flow-based approaches have begun to prove their superior capabilities over complex molecular datasets, succeeding in learning the molecular distribution and generating unique molecules with desired properties. On the other hand, the underlying operations of such models are not inherently interpretable by a human user, due to their highly complex and non-linear nature. In order to shed some light on the mysterious black-box nature of the generative models, a handful of research works have been published with a focus on the disentanglement aspect of the synthesized latent space by the generative models. However, most of the work relies on the knowledge of the ground truth factors, such as synthetic image datasets where each data point is calculated from a known set of latent generative factors. Moreover, the sub-category of explainability on molecular generation is a barren landscape, since most of the work is not directly translatable due to the biochemical domain-specific constraints such as the activity cliff phenomena and the bond and valence constraints.

In line with that, the experimentations carried out using various architectures that are trained with a benchmark dataset have revealed a fact: Simplifying assumptions on synthesized latent space do not always hold in the case of the drug generative model framework, partly due to model-paradigm related difficulties and ultimately due to the implementation domain's intricacies. In contrast, regardless of whether the assumptions over the latent space hold or not, the model can still perform reasonably well under entangled representations, from the perspective of the capability of synthesizing valid molecules. On the other hand, experiments prove that the concepts of navigation in the synthesized chemical space and steerability in terms of manipulation of desired drug properties are possible, although the amount of achieved success highly depends on the generative model paradigm and molecule representation choices. As a further direction, the experimentations in this thesis are yet to be enriched with different generative modeling paradigms, such as attention-based novel generative approaches and molecular properties that are more complex than commonly reported alternatives.

The explainability concept in drug generative modeling is still in need of overcoming technical difficulties that hinder the extent of the achievable disentanglement. Although current methods are still in their infancy, recent interpretability approaches are definitely serving as good starting points, and the recent period has been quite fruitful. As the utilization of the generative paradigm in the pharmaceutical industry becomes more prevalent, the needs for understanding the inner workings of such models and the achievement of transparency to make informed decisions become more relevant.

Bibliography

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan, 2017.

J. Avorn. The $2.6 billion pill—methodologic and policy considerations. *New England Journal of Medicine*, 372(20):1877–1879, 2015.

Y. Bian and X.-Q. Xie. Generative chemistry: drug discovery with deep learning generative models. *Journal of Molecular Modeling*, 27(3), feb 2021. doi: 10.1007/s00894-021-04674-8. URL `https://doi.org/10.1007%2Fs00894-021-04674-8`.

R. J. Campello, D. Moulavi, and J. Sander. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part II 17*, pages 160–172. Springer, 2013.

R. T. Q. Chen, X. Li, R. Grosse, and D. Duvenaud. Isolating sources of disentanglement in variational autoencoders, 2019.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

L. Dinh, D. Krueger, and Y. Bengio. Nice: Non-linear independent components estimation, 2015.

Y. Du, T. Fu, J. Sun, and S. Liu. Molgensurvey: A systematic survey in machine learning models for molecule design, 2022.

Y. Du, X. Liu, N. M. Shah, S. Liu, J. Zhang, and B. Zhou. Chemspace: Interpretable and interactive chemical space exploration. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL `https://openreview.net/forum?id=C1Xl8dYCBn`.

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

M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks, 2014.

J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998. doi: 10.1109/5254.708428.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=Sy2fzU9gl`.

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

W. Jin, R. Barzilay, and T. Jaakkola. Hierarchical generation of molecular graphs using structural motifs, 2020.

D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions, 2018.

D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2022.

N. Kokhlikyan, V. Miglani, M. Martin, E. Wang, B. Alsallakh, J. Reynolds, A. Melnikov, N. Kliushkina, C. Araya, S. Yan, and O. Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020.

G. Landrum. Rdkit: Open-source cheminformatics software. 2016. URL `https://github.com/rdkit/rdkit/releases/tag/Release_2016_09_4`.

J. Lucas, G. Tucker, R. Grosse, and M. Norouzi. Don't blame the elbo! a linear vae perspective on posterior collapse, 2019.

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

L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

J. Meyers, B. Fabian, and N. Brown. De novo molecular design and generative models. *Drug Discovery Today*, 26(11):2707–2715, 2021. ISSN 1359-6446. doi: https://doi.org/10.1016/j.drudis.2021.05.019. URL `https://www.sciencedirect.com/science/article/pii/S1359644621002531`.

E. N. Muratov, J. Bajorath, R. P. Sheridan, I. V. Tetko, D. Filimonov, V. Poroikov, T. I. Oprea, I. I. Baskin, A. Varnek, A. Roitberg, O. Isayev, S. Curtalolo, D. Fourches, Y. Cohen, A. Aspuru-Guzik, D. A. Winkler, D. Agrafiotis, A. Cherkasov, and A. Tropsha. Qsar without borders. *Chem. Soc. Rev.*, 49:3525–3564, 2020. doi: 10.1039/D0CS00098A. URL `http://dx.doi.org/10.1039/D0CS00098A`.

D. J. Newman and G. M. Cragg. Natural products as sources of new drugs from 1981 to 2014. *Journal of natural products*, 79(3):629–661, 2016.

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

D. Polykovskiy, A. Zhebrak, B. Sanchez-Lengeling, S. Golovanov, O. Tatanov, S. Belyaev, R. Kurbanov, A. Artamonov, V. Aladinskiy, M. Veselov, A. Kadurin, S. Johansson, H. Chen, S. Nikolenko, A. Aspuru-Guzik, and A. Zhavoronkov. Molecular sets (moses): A benchmarking platform for molecular generation models, 2020.

J.-L. Reymond and M. Awale. Exploring chemical space for drug discovery using the chemical universe database. *ACS chemical neuroscience*, 3(9):649–657, 2012.

P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL `https://www.sciencedirect.com/science/article/pii/0377042787901257`.

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

R. M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview, 2019.

C. Shi, M. Xu, Z. Zhu, W. Zhang, M. Zhang, and J. Tang. Graphaf: a flow-based autoregressive model for molecular graph generation, 2020.

A. Singh and T. Ogunfunmi. An overview of variational autoencoders for source separation, finance, and bio-signal applications. *Entropy*, 24(1), 2022a. ISSN 1099-4300. doi: 10.3390/e24010055. URL `https://www.mdpi.com/1099-4300/24/1/55`.

A. Singh and T. Ogunfunmi. An overview of variational autoencoders for source separation, finance, and bio-signal applications. *Entropy*, 24(1):55, 2022b.

J. M. Stokes, K. Yang, K. Swanson, W. Jin, A. Cubillos-Ruiz, N. M. Donghia, C. R. MacNair, S. French, L. A. Carfrae, Z. Bloom-Ackermann, V. M. Tran, A. Chiappino-Pepe, A. H. Badran, I. W. Andrews, E. J. Chory, G. M. Church, E. D. Brown, T. S. Jaakkola, R. Barzilay, and J. J. Collins. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702.e13, 2020. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2020.01.021. URL `https://www.sciencedirect.com/science/article/pii/S0092867420301021`.

D. Stumpfe, H. Hu, and J. Bajorath. Evolving concept of activity cliffs. *ACS omega*, 4(11):14360–14368, 2019.

F. Trozzi, X. Wang, and P. Tao. Umap as a dimensionality reduction tool for molecular dynamics simulations of biomacromolecules: a comparison study. *The Journal of Physical Chemistry B*, 125(19):5022–5034, 2021.

L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

D. Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005. URL `https://doi.org/10.1021/ci00057a005`.

J. Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9(1):36–45, jan 1966. ISSN 0001-0782. doi: 10.1145/365153.365168. URL `https://doi.org/10.1145/365153.365168`.

Y. Yacoby, W. Pan, and F. Doshi-Velez. Failure modes of variational autoencoders and their effects on downstream tasks, 2022.

C. Yan, S. Wang, J. Yang, T. Xu, and J. Huang. Re-balancing variational autoencoder loss for molecule sequence generation, 2020.

C. Yang, Y. Shen, and B. Zhou. Semantic hierarchy emerges in deep generative representations for scene synthesis, 2020.

C. Zang and F. Wang. MoFlow: An invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery &amp Data Mining*. ACM, aug 2020. doi: 10.1145/3394486.3403104. URL `https://doi.org/10.1145%2F3394486.3403104`.

X. Zeng, F. Wang, Y. Luo, S. gu Kang, J. Tang, F. C. Lightstone, E. F. Fang, W. Cornell, R. Nussinov, and F. Cheng. Deep generative molecular design reshapes drug discovery. *Cell Reports Medicine*, 3(12):100794, 2022. ISSN 2666-3791. doi: https://doi.org/10.1016/j.xcrm.2022.100794. URL `https://www.sciencedirect.com/science/article/pii/S2666379122003494`.

Appendix A

Model Hyperparameters

Model hyperparameters of the baseline VAE model are given in table A.1

Parameter	Value
q_cell	GRU
q_bidir	False
q_d_h	256
q_n_layers	1
q_dropout	0.5
d_cell	GRU
d_n_layers	3
d_dropout	0
d_z	128
d_d_h	512
freeze_embeddings	False

Table A.1: Baseline VAE parameters

From top to bottom, the hyperparameters correspond to: Encoder's RNN cell type, a flag to specify whether the encoder RNN is bidirectional or not, the encoder RNN's hidden dimensionality, the number of layers for the encoder's RNN, the dropout rate for encoder layers, decoder's RNN cell type, number of RNN layers in the decoder, the dropout rate for decoder layers, the dimensionality of the latent space, the hidden size of the decoder's RNN, a flag specifying whether the embeddings should be fixed or not respectively.

Model hyperparameters of the HierVAE model are provided in table A.2

Parameter	Value
rnn_type	LSTM
hidden_size	250
embed_size	250
latent_size	32
depthT	15
depthG	15
diterT	1
diterG	3
dropout	0.2
mask_row_size_list	1

Table A.2: HierVAE parameters

From top to bottom, the specified parameters correspond to: the type of RNN to be used in Message Passing Network, the size of the linear layer and the size of the node feature vector, embedding size for nn.Embedding, hidden layer number of RNN of the Message Passing Network for tree representation, hidden layer number of RNN of the Message Passing Network for graph representation, depth of RNN of MPN for tree and intermediate representation for the decoder, and the dropout rate during training respectively.

The model hyperparameters for the MoFlow model are given in Table A.3

Parameter	Value
b_n_flow	10
b_n_block	1
b_hidden_ch	128,128
b_conv_lu	1
a_n_flow	27
a_n_block	1
a_hidden_gnn	64
a_hidden_lin	128,64
mask_row_size_list	1
mask_row_size_list	1
dropout	0.2

Table A.3: MoFlow parameters

From top to bottom, the hyperparameters correspond to: the number of masked glow coupling layers per block for the bond tensor, the number of glow blocks for the bond tensor, the hidden channel list for the bonds tensor and the delimited list input, the number of masked flow coupling layers per block for atom matrix, the hidden dimension list for graph convolution for atoms matrix and the delimited list input, the hidden dimension list for linear transformation for atoms and delimited list input, mask row size list for atom matrix and delimited list input, and mask row stride list for atom matrix, delimited list input, and dropout rate during the training respectively.

Appendix B

Latent Space Visualization with t-SNE

The latent space visualization with t-SNE with $\max \beta = 0.1$ at the top and $\max \beta = 1.0$ at the bottom. The perplexity values are 40, 100, 200 from left to right respectively.

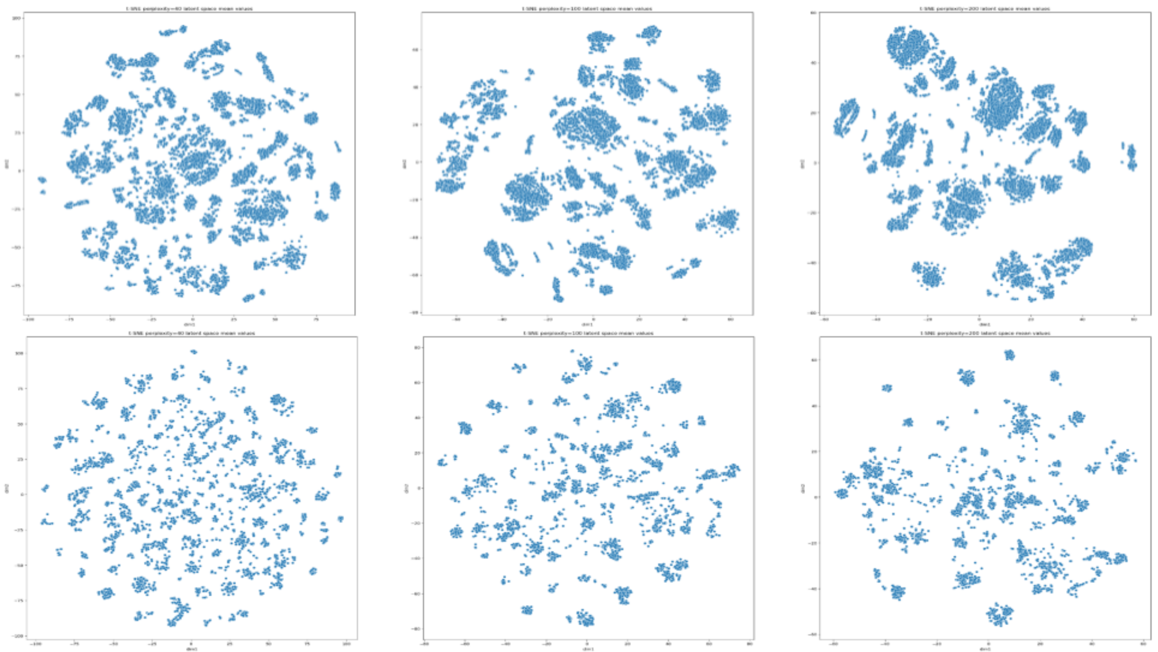


Figure B.1: $\max \beta = 0.1$(top) and $\max \beta = 1$(bottom) latent space with t-SNE

Appendix C

Property Clustering Results

The clustering results with UMAP reduced space for BCUT2D descriptors are at Figure C.1. The obrained silhouette score average is 54.51% with 3 clusters. Among the experimented parameter ranges of [2-50] for minimum cluster sizes and [1-40] for minimum samples for HDBSCAN algorithm, the highest score is obtained with minimum cluster size of 50 and minimum samples of 5.

The clustering of Lipinski descriptors in the UMAP reduced space is provided in Figure C.2. The obtained silhouette score is 63.79% with 48 clusters. Highest average silhouette score is obtained by minimum cluster sizes of 50 and minimun number of samples 20 for HDBSCAN.

For the Topological descriptor clustering in the UMAP reduced space, a silhouette coefficient score of 30% is obtained as the highest with the hyperparameter values of minimum, cluster size of 8 and minimum samples of 5. There are 639 clusters detected in total, which seems not to be very precise given the lowest silhouette score obtained among all other property clustering experiments. The clustering is illustrated in Figure C.3

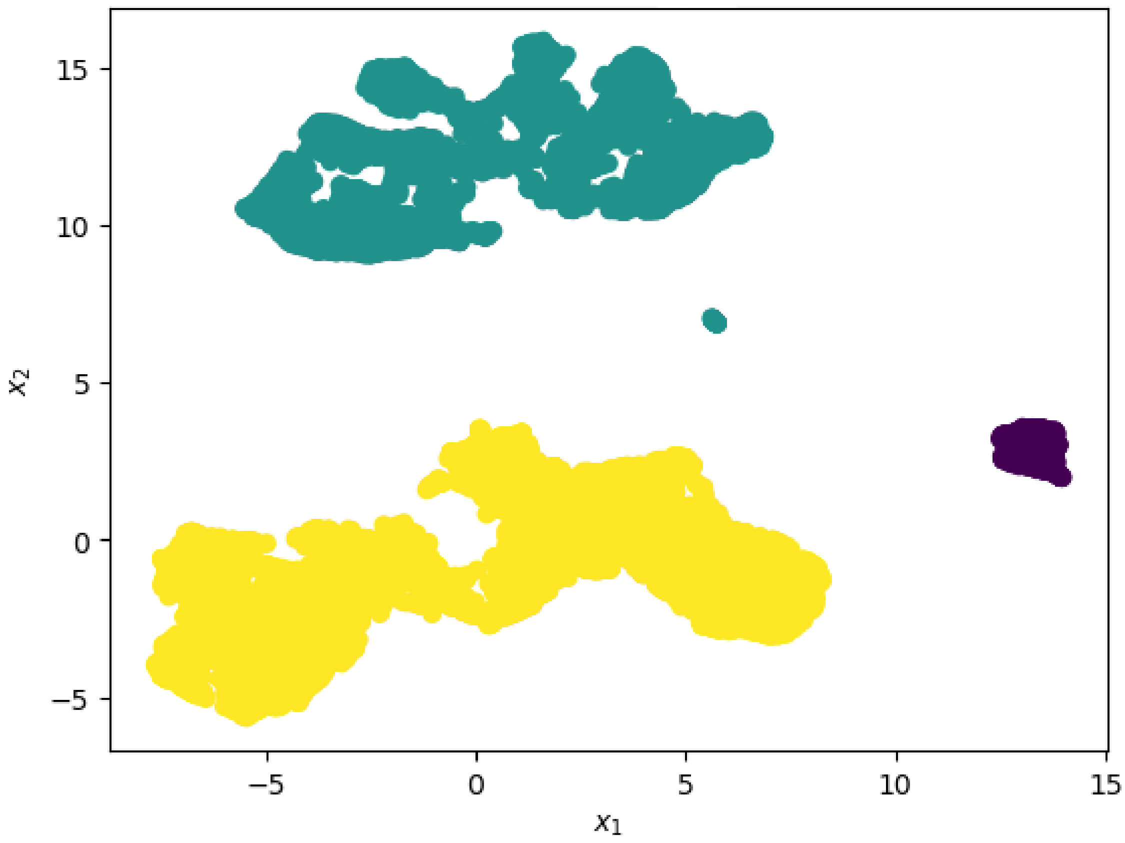


Figure C.1: BCUT2D property clustering in UMAP reduced space

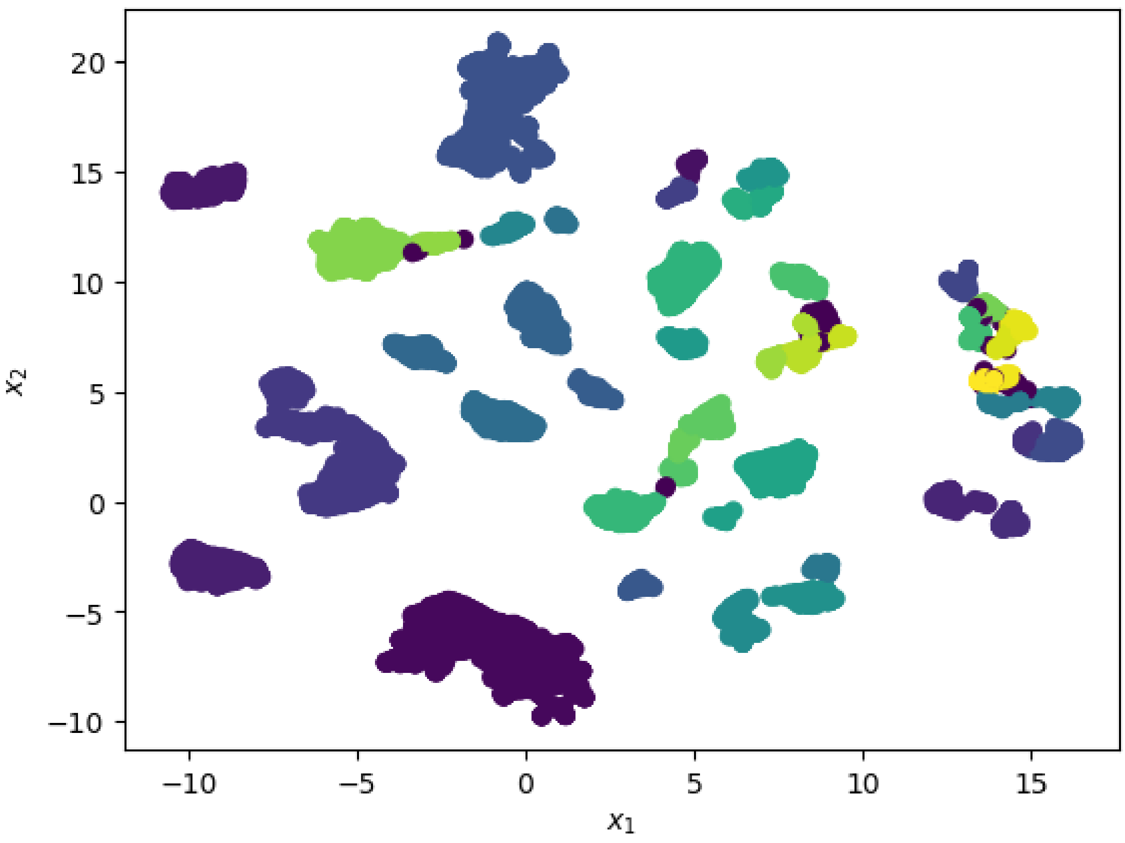


Figure C.2: Lipinski property clustering in UMAP reduced space

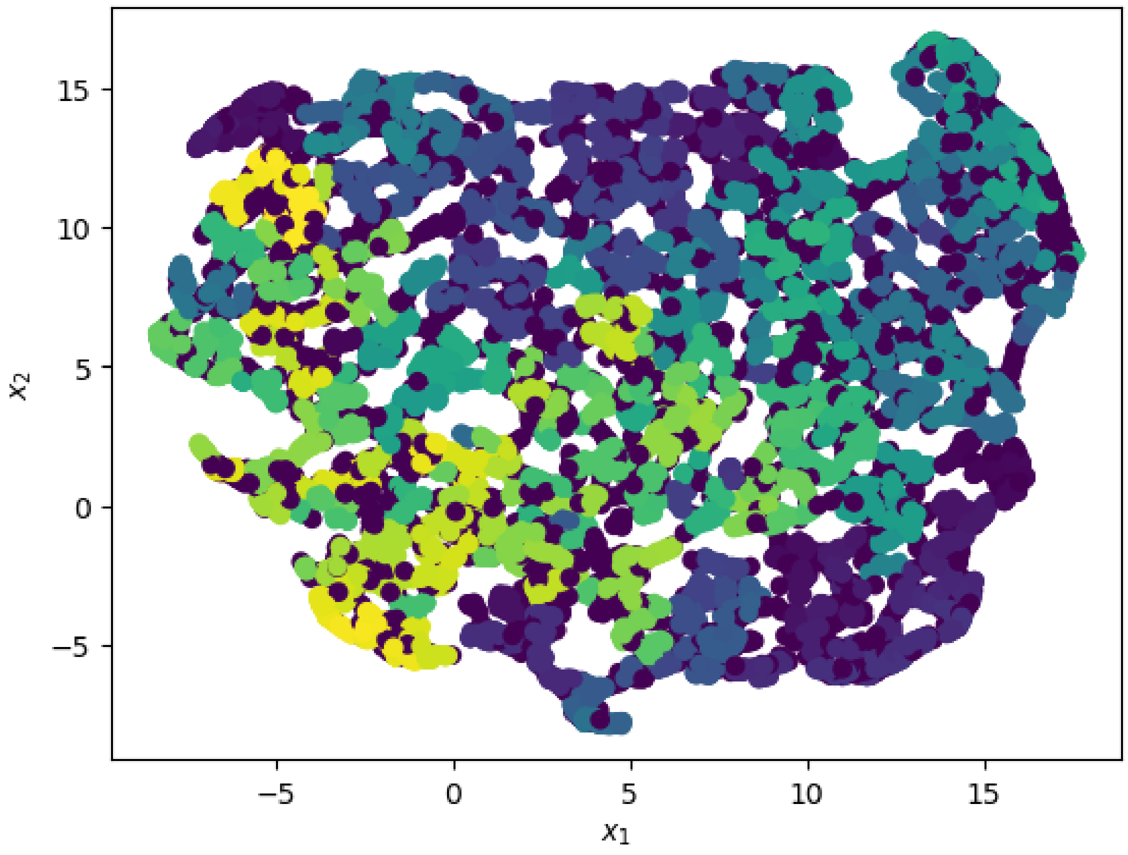


Figure C.3: Topological property clustering in UMAP reduced space

Appendix D

ChemSpacE Singular Property Descriptor Values of SSR and RSR-G Variants

Per feature scores obtained for SSR, RSR-G (0.05) and RSR-G (0.1) are provided in tables D.1, D.2 and D.3 respectively. For the SSR, baseline VAE failed to obtain successful extrapolations, whereas HierVAE configurations are capable of produce strict extrapolations, although the achieved success varies between different property descriptors. Overall, Chi1n and BalabanJ obtained the lowest scores, while MolWt, MolMr and HeavyAtomCount descriptors lead to the best results. In the case of MoFlow, highest strict exrapolation success is obtained by MolWt and the lowest by MolMr descriptor.

On the other hand, the allowance of 5% over the global property ranges resulted in a drastical increase in the obtained successful manipulations, especially in the HierVAE variants. Although the success rate for baseline models also increased, for the most properties the value is still less than 1%. The relative increase in success over the strict case is not as high as HierVAE for the MoFlow model.

Most notably, the allowance of 10% over the global manipulation rates enabled baseline model to generate successful extrapolations for certain property descriptors. However, such a big leap from a success rate of around $\%1$ to almost 72% indicate that such a large relaxation compromises the monotonicity trend, causing too large deviations in consecutive manipulations to be accepted as successful.

Model	QED	BertzCT	BalabanJ	Chi1n	HeavyAtomCount	MolMR	MolLogP	MolWt
MOSES VAE max β=0.1	0.0	0.0	0.001	0.0	0.002	0.0	0.001	0.001
MOSES VAE max β=1.0	0.0	0.0	0.0	0.0	0.0	0.001	0.001	0.001
HierVAE max β=0.1	0.202	0.289	0.114	0.099	0.327	0.254	0.215	0.304
HierVAE max β=1.0	0.173	0.22	0.127	0.123	0.214	0.252	0.18	0.273
MoFlow	0.65	0.625	0.695	0.655	0.755	0.595	0.635	0.71

Table D.1: ChemSpacE results of SSR per feature value

Model	QED	BertzCT	BalabanJ	Chi1n	HeavyAtomCount	MolMR	MolLogP	MolWt
MOSES VAE max β=0.1	0.008	0.008	0.017	0.007	0.029	0.01	0.005	0.013
MOSES VAE max β=1.0	0.01	0.009	0.015	0.008	0.025	0.012	0.01	0.017
HierVAE max β=0.1	0.43	0.75	0.37	0.72	0.754	0.738	0.739	0.82
HierVAE max β=1.0	0.386	0.595	0.393	0.696	0.702	0.807	0.586	0.78
MoFlow	0.715	0.705	0.76	0.735	0.765	0.675	0.705	0.785

Table D.2: ChemSpacE results of RSR-Global with $\epsilon = \delta = 0.05$ per feature value

Model	QED	BertzCT	BalabanJ	Chi1n	HeavyAtomCount	MolMR	MolLogP	MolWt
MOSES VAE max β=0.1	0.205	0.08	0.323	0.371	0.563	0.641	0.429	0.15
MOSES VAE max β=1.0	0.23	0.065	0.347	0.414	0.642	0.716	0.496	0.159
HierVAE max β=0.1	0.632	0.895	0.648	0.895	0.88	0.9	0.909	0.913
HierVAE max β=1.0	0.572	0.821	0.657	0.886	0.898	0.926	0.839	0.901
MoFlow	0.745	0.73	0.765	0.755	0.775	0.69	0.755	0.8

Table D.3: ChemSpacE results of RSR-Global with $\epsilon = \delta = 0.1$ per feature value