Towards automated molecular search in drug space

by

Rustam Zhumagambetov

B.S., Nazarbayev University (2019)

Submitted to the Department of Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

at the

NAZARBAYEV UNIVERSITY

Apr 2021

Author .....
Department of Computer Science
April 27, 2021

Certified by .....
Siamac Fazli
Associate Professor
Thesis Supervisor

Certified by .....
Vsevolod A. Peshkov
Assistant Professor
Thesis Supervisor

Accepted by .....
Vassilios D. Tourassis
Dean, School of Engineering and Digital Sciences

Publications

Rustam Zhumagambetov, Daniyar Kazbek, Mansur Shakipov, Daulet Maksut, Vsevolod A. Peshkov, and Siamac Fazli. cheml.io: an online database of ml-generated molecules. *RSC Adv.*, 10:45189–45198, 2020. Published.

Rustam Zhumagambetov, Vsevolod A. Peshkov, and Siamac Fazli. Transmol: Repurposing Language Model for Molecular Generation. 4 2021. Preprint.

Towards automated molecular search in drug space

by

Rustam Zhumagambetov

Submitted to the Department of Computer Science
on April 27, 2021, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science

Abstract

Recent advances in convolutional neural networks have inspired the application of deep learning to other disciplines. Even though image processing and natural language processing have turned out to be the most successful, there are many other areas that have benefited, like computational chemistry in general and drug design in particular. From 2018 the scientific community has seen a surge of methodologies related to the generation of diverse molecular libraries using machine learning. The first goal is to provide an accessible way of using machine learning algorithms to chemists without technical knowledge. Hence, cheML.io, a web database that contains virtual molecules generated by 10 recent ML algorithms, is proposed. It allows users to browse the data in a user-friendly and convenient manner. ML-generated molecules with desired structures and properties can be retrieved with the help of a drawing widget. For the case of a specific search leading to insufficient results, users are able to create new molecules on demand. The second goal is to develop an algorithm that allows the generation of diverse focused libraries utilizing one, or two seed molecules which guide the generation of *de novo* molecules. Here a variant of transformers, an architecture recently developed for natural language processing, was employed for this purpose. The results indicate that this model is indeed applicable for the task of generating focussed molecular libraries and leads to statistically significant increases in some of the core metrics of the MOSES benchmark. A benchmark that provides baselines and metrics that can characterize the main attributes of the algorithms by examining the generated molecules. In addition, a novel way of generating libraries where two seed molecules can be fused is introduced.

Thesis Supervisor: Siamac Fazli
Title: Associate Professor

Thesis Supervisor: Vsevolod A. Peshkov
Title: Assistant Professor

Acknowledgments

While the journey was a relatively short one, only two years, it was especially fruitful for me as I was able to engage in an amazing interdisciplinary project.

I would like to thank my first supervisor, Professor Siamac Fazli, who initiated the project and hired me, providing invaluable experience, whose expertise in machine learning and research has molded me into an aspiring scientist. Observing your thought process during the execution of the project enabled my professional growth.

I would equally like to thank my second supervisor, Professor Vsevolod A. Peshkov, whose striving for perfection has demonstrated to me the highest standard of rigor in research. Your timely suggestions and pieces of advice, offered along the way, have shaped this thesis and brought my work to a higher level.

Your commitment to the research during pandemic and disruption greatly inspired me and demonstrated to me that research projects can be conducted despite closed borders and across different timezones.

I would also like to thank an external committee member, whose name has not been disclosed yet to me, for agreeing to participate in my defense.

Grant provided by Young Researchers Alliance under their Fostering Research and Innovation Potential Program has partially supported my research and eased my "universitysickness" during self-isolation.

I would also like to thank my family and friends who have been with me throughout the studies, offering support and counseling.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chemistry is frequently referred to as a "central science" for its key role in advancing technological progress and human well-being through the design and synthesis of novel molecules and materials for energy, environmental, and biomedical applications.

Medicinal chemistry is a highly interdisciplinary field of science that deals with the design, chemical synthesis, and mechanism of action of biologically active molecules as well as their development into marketed pharmaceutical agents (i.e. drugs). The creation of new drugs is an incredibly hard and arduous process. One of the key reasons being the fact that the 'chemical space' of all possible molecules is extremely large and intractable. Even though it is estimated that the chemical space of molecules with pharmacological properties is in the range of $10^{23} - 10^{60}$ compounds [36], this order of magnitude leaves the work of finding new drugs outside the reach of manual labor.

In general, medicinal chemists need to determine molecules that are active and selective towards specific biological targets to cure a particular disease while keeping the risks of negative side effects minimal. As the number of molecules that require testing to identify an ideal drug candidate constantly increases, it raises the overall cost of the drug discovery process. Therefore, the need for algorithms that are able to narrow down and optimize these efforts has recently emerged. Specifically, computer algorithms can assist with creating new virtual molecules as well as performing conformational analysis [2, 21] and molecular docking [11, 32] to determine the affin-

ity of novel and known molecules towards specific biological targets. Aside from the generation of *de novo* molecules researchers could run a search on the database of the known molecular compounds with desirable structural queries[45].

Therefore this thesis proposes a database of virtual molecules generated by 10 machine learning algorithms, and a novel application of pure attention architecture, Transformer, repurposed for the *de novo* molecule generation.

A database is aimed to help chemists to take advantage of the novel computational approaches for the generation of novel molecules in an accessible fashion. It is possible to search this database using similarity and substructure queries. In case the query leads to suboptimal results, the users are able to create new molecules on demand.

A novel application of attention mechanisms, a model that takes advantage of the nature of SMILES as a construct with its own grammar is created to generate focused molecular libraries. Such libraries can be used by chemists in their drug screening campaigns. To evaluate the results the MOSES benchmark [34] has been used, which allows the juxtaposition of already existing and future methods.

I believe that the proposed work will be helpful in the field of drug design, where novel molecules are used for the creation of life-saving drugs.

Chapter 2

Literature Review

2.1 Representations of molecules

When beginning discussion of automating molecular search a natural question how molecules, a physical collection of atoms that are arranged in 3D space, can be represented.

2.1.1 In communications

Since the beginning of the chemical discourse a need arised to describe the subjects of discussions. Usually, the formula contains all atoms that constitute the molecule. For instance, $C_8H_{10}N_4O_2$. However, such formulae, being unambiguous, have a serious flaw – as the length of the formula grows, it becomes harder to pronounce. So the International Union of Pure and Applied Chemistry (IUPAC) has created a standard that preserves the advantages of the chemical formula and eliminates its disadvantages. Some of the molecules become so popular that they receive their own special name, for example, $C_8H_{10}N_4O_2$ – 1,3,7-Trimethylpurine-2,6-dione, or caffeine.

2.1.2 In silico

As chemists that began discussing molecules invented new notations, the emergence of computer-aided chemical research required appropriate digital representation of

molecules. Currently, two approaches are commonly used: representations as a string, and as a graph (See Figure 2-1).

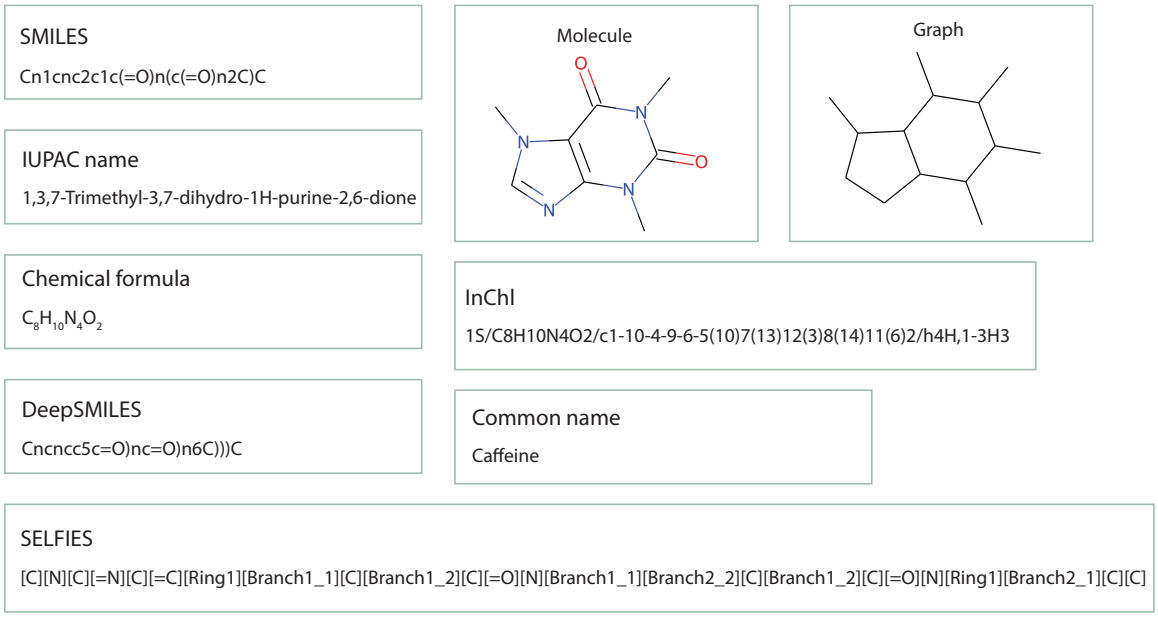


Figure 2-1: The overview of molecular representations

Molecules as strings

Wiswesser Line Notation (WSN) is a chemical notation proposed by Wiswesser in the 1950s. The primary objectives of this notation are ease of use, expressiveness (capability of handling both inorganic and organic molecules), and recognizability after typing.

In the 1980s simplified molecular input line entry system (SMILES) specification had emerged, aimed to create a molecular encoding that is computationally efficient and human readable [49]. The original encoding is based on 2D molecular graphs. Intended application areas are fast and compact information retrieval and storage.

SYBYL Line Notation (SLN) is a specialized chemical notation, inspired by SMILES. In addition to expressing molecules SLN also encodes substructure queries. Such extension has found its use in chemical databases. The compact size of the resulting strings lowers the strain on storage and networking.

DeepSMILES is a more recent addition to the collection of notations. As the

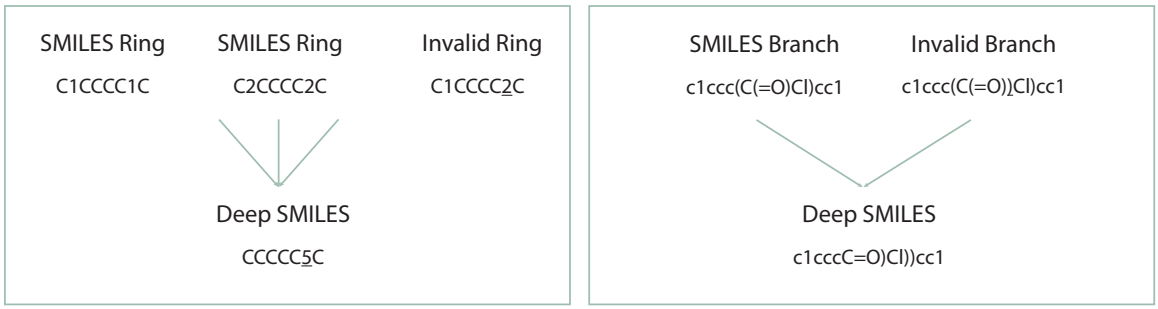


Figure 2-2: The demonstration of two common patterns occurred in invalid SMILES generated by machine learning algorithms

name suggests the goal of DeepSMILES is to improve the results of deep neural architectures. O'Boyle and Dalke noticed the patterns of failed (invalid) results of machine learning algorithms - unbalanced parenthesis and ring closures, see Figure 2-2. To eliminate the possibility of failure, DeepSMILES proposes to simplify the notation: instead of ring closures a single number is used, and opening parenthesis is replaced by a collection of closing parenthesis, where the number of closing parenthesis represents the depth of branch, see Figure 2-2 for examples.

The latest state-of-the-art representation used for machine learning is SELFIES. As well as DeepSMILES, SELFIES improves the machine learning algorithms by eliminating the need to learn a complex syntax of SMILES. However, Krenn et al. have further improved the representation to be resistant to random string mutations, to achieve that any SELFIES string can be mapped to the molecule. The consequence of 100% valid strings is that internal workings of generative machine learning algorithms can be easily investigated.

Molecules as graphs

The resemblance of molecules and graphs has been studied at least since the 1960s [5]. Atoms are represented as vertices and bonds are the edges between them. Problems like relationships between molecular structures and properties have been studied in the intersection of graph theory and chemistry. A common approach would be the translation of molecules atom by atom.

In 2018, a surge of graph-based deep learning algorithms has appeared [50]. A

rapid development has allowed a detailed scrutiny of the graph representations. One of the discovered problems of the naive approach would be chemically invalid intermediaries of graph-based machine learning algorithms. So extensions have been proposed. For example, Jin et al. [23] proposed a method for encoding and decoding the molecules into graphs in two phases. In the first step, a junction tree is extracted from a given molecule and functions as a scaffold of subgraph components. These components are then used as building blocks for creating new molecules.

2.2 Early drug discovery

Early approaches to drug discovery were focused in 2 directions: combinatorial generation of molecules, and matching a specific structure to the database of known molecular compounds.

2.2.1 Combinatorial approach

LUDI [7] is a computer program whose objective is to produce an enzyme inhibitor by examining a 3D structure of the target protein. To do so, the program was using a library of 600 possible fragments that could be potential parts of the final novel molecule. If the fragment can fit the target, meaning it can interact with the protein, then the fragment is suitable to be a part of the result. In the final step all or some suitable fragments are connected together using bridge fragments.

HOOK [12] attempts to improve upon LUDI. While LUDI uses multicopy simulation search (MCSS) as a primary tool for fitting the fragments, HOOK relies on heuristic rules defined by experts. Usage of heuristics drastically improves computational speed and allows the generation to occur in real-time.

Another combinatorial approach to molecule generation is through molecular evolution. For instance, SPROUT [15] builds molecules in steps using a genetic algorithm on molecular graphs. To accelerate the generation, a number of expert-crafted heuristics are used. Unlike previous approaches SPROUT does not require a library of fragments.

A more recent work that employs a combinatorial approach is the fragment-based molecular evolution algorithm [25]. The seed molecule is used as an input to generate similar molecules. Tanimoto distance, a measure of how one molecular fingerprint(a bitstring representation of molecular structure) is similar to other between 0 and 1, is used to measure similarity. The seed molecules is then dissected into seed fragments. Expert-crafted connection rules are then employed for crossover of the fragments. The advantage of this method is that resulting molecules differ from the seed molecule not only in side-chains, but also in scaffolds.

2.2.2 Databases of chemical compounds

ALLADIN [45] is a computer program that combines several algorithms for searching using geometric, steric, and substructural criteria. While previous approaches aim to generate novel molecules, ALLADIN attempts to reuse massive databases of molecules through automatic screening of existing compounds for specific biological properties.

Brint and Willet [6] describe an algorithm for substructure queries into a database of 3D chemical compounds.

2.3 Brief introduction to machine learning

Machine learning is a discipline that studies methods for creation of mathematical models from data. While conventional models harness predictive power that usually requires expert knowledge, machine learning models are data-driven and can learn hidden patterns from data. With an increasing volume of data available machine learning algorithms become an inevitable tool for data processing and intelligence extraction.

The earliest attempts to machine learning could be attributed to the Rosenblatt's perceptron in 1958 [38], where a construct mimicking a single neuron's brain cell, was used for prediction. Learning from data required an optimization algorithm. The learning rule is to update weight only when misclassifcation occurs. To increase the predictive power of the perceptron researcher have been increasing the number lay-

ers. The approach have been called a multilayer perceptron [22]. Then an algorithm for backpropagation, calculation of the gradient on any layer of the multilayer perceptron, has appeared [39]. Using old gradient descent [9], an interative method for optimization of objective function (finding minimum of the loss function in MLP) with the following rule:

$$w_{i+1} = w_i - \eta \nabla Q(w_i) \tag{2.1}$$

where η is a learning rate and Q is a loss function. Coupled with the gradient descent the backpropagation is one of the cornerstones of neural algorithms. However, the main breakthrough happened only in 2010s when hardware's performance could sustain application of aforementioned techniques for the "big data".

2.3.1 Machine learning for language modelling

Language modelling tries to predict the next word given a sequence of earlier words. It is a task that is found whenever textual information is encountered: speech recognition, machine translation, and image captioning. Given that SMILES strings can be used as molecular representation, the language modelling task can be extended to the domain of molecular modelling.

2.3.2 SMILES as a sentence

In 2014 Cadeddu et al. [8] demonstrated that natural language and organic molecules have similar distribution of text and molecule fragments. The performed analysis implied that methods of computational linguistics, or language modelling can be used for modelling of the molecular domain.

2.4 The Advent of deep learning

Generative machine learning frameworks for the creation of molecular libraries can be roughly classified into three categories and are based on autoencoders [27], recurrent neural networks (RNNs) [39] and generative adversarial networks (GANs) [18].

2.4.1 GAN-based algorithms

GAN is a novel deep learning architecture proposed by Goodfellow et al. [18]. It consists of two neural networks that are trained simultaneously: the generative one that takes noise and produces output, and the discriminator one that attempts to classify the output as real or fake. Through the adversary the generator improves to produce a realistic output, be it images, or any other kind of the information, including SMILES strings.

2.4.2 Autoencoder-based algorithms

The idea of autoencoders can be traced back to the 80s [27, 17]. Autoencoders are neural networks that consist of an encoder and a decoder that tries to reconstruct the input. The encoder transforms the input into a compressed vector representation and the decoder attempts to recover the input from this compressed representation. A hidden layer with a limited number of nodes between the encoder and decoder represents the minimal amount of information that is needed to decode the original input. Such architectures can be used for denoising, dimensionality reduction and have more recently also been applied for drug discovery [20].

2.4.3 RNN-based algorithms

Recurrent neural networks have been studied for more than 30 years [39]. RNNs consist of several nodes that form a directed graph. In addition to processing input, they also receive their earlier outputs as an input. The output is, therefore, *recurring* as input in every time step. Applications of RNNs encompass data domains, where input data is "sequentially connected", like natural language processing, music generation, text translation, automatic generation of image captions, among others.

2.4.4 Attention-based algorithms

The attention mechanisms in machine learning were created in an attempt to mimic cognitive attention, based on the observation that humans concentrate selectively. So

attention decides which part of the input information is important. To determine the importance, the gradient descent is used.

Chapter 3

cheML.io: an online database of ML-generated molecules [1]

3.1 Introduction

After a review of the current literature, it became apparent that although there is a growing number of machine learning algorithms for molecular generation, there is no accessible way to use them for chemists. Hence, the goal is to replicate a number of recent ML algorithms for *de novo* molecular generation for comparison of their molecular outputs. The resulting molecules were unified into an online database of browse-able virtual molecules - cheML.io. The built-in drawing widget allows performing substructure and similarity searches. In case of unsatisfiable results, new molecules could be generated on demand.

3.2 ML algorithms

To populate the database with ML-generated virtual molecules, several machine learning frameworks have been implemented. As mentioned previously, existing machine learning methods for molecular generation can be roughly divided into three major

[1] This chapter exclusively consists of the author's contribution to [53]. Reproduced from [53] with permission from the Royal Society of Chemistry.

categories: GAN-based methods, autoencoder-based methods, and RNN-based methods. Excellent reviews providing a comprehensive analysis of all available methods to date have recently appeared in the literature [40, 13]. Below can be found a brief description of several machine learning frameworks that were utilized to populate the database of ML-generated molecules. A graphical representation of all considered methodologies, namely Objective Reinforced Generative Adversarial Network (ORGAN) [19], Objective Reinforced Generative Adversarial Network for Inverse-Design chemistry (ORGANIC) [41], Conditional Diversity Network (CDN) [20], Variational Autoencoder with Multilayer Perceptron (ChemVAE) [16], Grammar Variational Autoencoder (GrammarVAE) [26], Conditional Variational Autoencoder (CVAE) [28], Recurrent Neural Networks (RNN) [42], Junction Tree Variational Autoencoder (JT-VAE) [23], a CycleGAN [52] based model (MolCycleGAN) [30] and Semi Supervised Variational Autoencoder (SSVAE) [24].

3.2.1 Autoencoder-based methods

Conditional Diversity Network is a deep learning network that utilizes a variational auto-encoder (VAE) with a "diversity" layer to generate molecules that are similar to the prototype, yet different. To do this they introduce the diversity layer to the vanilla VAE architecture. So, during the generation stage instead of using random noise as an input to the decoder, the CDN uses a sample of the prototype as an input to the decoder. This allows sampling molecules that have similar features when compared to the prototype.

ChemVAE is an autoencoder with a multilayer perceptron that is used for property prediction. While the autoencoder is employed to learn the latent space of valid molecules, the multilayer perceptron is used to generate molecules with desired properties. Trained jointly with the autoencoder, the perceptron organizes the latent space, grouping the molecules with similar property values in the same latent region. This allows the sampling of molecules with desired properties.

While methods like ORGAN or ORGANIC use a GAN and RNN to generate sequence data like the SMILES molecule representation, GrammarVAE attempts to

avoid learning the syntax of the SMILES. Instead of having a GAN learn the syntax of the SMILES format, GrammarVAE uses the fact that SMILES can be represented by context-free grammar and learns the grammar rules, thus avoiding the generation of molecules that are syntactically invalid. To do this GrammarVAE calculates the parsing tree of a molecule, then converts it into one-hot-encoding, having the variational autoencoder learn the sequence of applied grammar rules, rather than individual characters.

JT-VAE deviates from the traditional approach to molecule generation. Instead of using the SMILES representation of the molecule, JT-VAE utilizes a direct graph representation of the molecule. JT-VAE builds new molecules by using subgraphs of the old ones. While other methods often use an atom combination approach, JT-VAE uses component combination. Combination of the graph representation and the variational autoencoder almost always yields valid molecules. CVAE is aimed to generate molecules with several desired properties. Earlier results indicate that optimization for one property may unintentionally change other properties. In order to counter this effect, CVAE is designed as an artificial neural network that is suitable for optimization of multiple properties. To achieve this goal CVAE uses a conditional vector for both encoding and decoding. Such an architecture allows for the incorporation of properties into the latent space.

3.2.2 RNN-based methods

In this method, RNN is represented by long short term memory (LSTM). The architecture is comprised of 3 stacked LSTM layers. To overcome the problem of generating unfocused molecules, transfer learning, process where pre-trained weights are used, is employed. After transfer learning, a small subset of the focused molecules is used to fine-tune the model, so that it generates molecules with desired properties.

SSVAE is a semi supervised model that has advantages when dealing with datasets where only a part of the dataset is labeled with properties. It consists of 3 bi-directional RNNs, which are used for encoding, decoding, and predicting. In this model, the property prediction and molecule generation are combined in a single

network. The novel molecules are decoded from the latent space, which is the product of the trained encoder.

3.2.3 GAN-based methods

ORGAN is an artificial neural network architecture based on SeqGAN [51], which is adapted for melody and molecule generation. It feeds SMILES molecules to the generative network of the GAN and uses Wasserstein-1 distance, a distance function between two distributions (also known as earth mover's distance), to improve the results of the training.

ORGANIC is a framework for the generation of novel molecules, which is based on ORGAN. It is a more chemistry oriented version of ORGAN. The limitation of the ORGANIC is that it has an unstable output of molecules. The range of invalid molecules that are created deviates between 0.2 and 99 percent.

MolCycleGAN is based on CycleGAN. To generate novel molecules with similar properties MolCycleGAN uses JT-VAE as a latent space producer, then utilizes GAN to produce the molecule. To feed the GAN, a molecule with desired features is used and the resultant latent space embedding is then transformed back to the new molecule with a similar structure and desired properties.

3.3 Database overview

3.3.1 Implementation of methods

Implementations of each method aside from RNN[2] were mentioned in the original publications. All of the algorithms were written in Python with the help of either Pytorch [33] or Tensorflow [1]. Training of all the methods except for RNN was conducted using the samples of molecules from ZINC that were provided by the authors of the original implementations. For RNN a ZINC-based training dataset, which was provided by the authors of JT-VAE [23] have been used. In addition,

[2]Implementation of RNN from https://github.com/LamUong/Generate-novel-molecules-with-LSTM was used

implementations of CDN and RNN methods were run utilizing a 1.6 million molecules sample of ChEMBL [31] as a training dataset. As a result, ca. 0.62 thousand out of the total 3.64 thousand molecules generated with CDN and ca. 0.65 million out of the total 0.96 million molecules generated with RNN were obtained based on the ChEMBL training data.

3.3.2 Molecule storage and preprocessing

All generated molecules along with their properties are stored in PostgreSQL [44], a free and open-source relational database management system with a variety of modules that allows to use them in different contexts. For instance, RDKit Cartridge enables efficient search across molecules.

Preprocessing is an important step in the management of a large molecular database, containing millions of members. Utilizing RDKit [35] 2.9 million molecules, produced by the above mentioned generative ML algorithms, were inserted into the database in canonical SMILES format. During the insertion, a fraction of molecules were discarded: 174000 were invalid (according to RDKit) and for 633 RDKit was not able to construct canonical SMILES. In addition, all duplicates were removed. In total, 2.8 million molecules were inserted along with computed properties that are listed in Lipinski's rule of five [29]. Moreover, other medicinal chemistry-relevant properties have been computed such as the number of rotatable bonds and the number of saturated rings.

The typical operations on databases composed of a large number of molecules involve searching by substructure and searching by similarity. To optimize such queries Morgan Fingerprints (Circular Fingerprints) have been precomputed as well as the RDKit implementation of Extended Connectivity Fingerprints [37].

3.4 Generation on demand

Currently, Conditional Diversity Networks [20] are employed for the generation of new molecules on demand. Based on the observations, the algorithm achieves the best

results when the training is performed for each requested SMILES input. Current approach can be summarized as follows:

1. Fetch molecules that are *similar* to the seed molecule from three databases: ZINC [43], ChEMBL [31] and cheML
2. Utilize these molecules as input data for the first training
3. Fetch molecules from the above databases that contain the seed molecule as a *substructure*
4. Utilize these molecules as input data for the second training
5. Combine previous input data and use them as input for a third training
6. Generate molecules using all three distinct models built by each of the above input data. Filter the resulting molecules based on their similarity score with the seed molecule. Exclude the molecules that are already present in the cheML.io database and those featuring the same structural backbone (i.e. different only by the stereochemical features) and send the outcome to the user by email
7. Add novel molecules to the cheML.io database

3.5 Results

Model	Architecture	Learning Technique	Molecule representation	Property targeting	Computational costs	Training Dataset size
JT-VAE	VAE	autoencoder	graph	yes	medium	250k
RNN	RNN	direct flow	SMILES	no	low	250k
GrammarVAE	VAE	autoencoder	SMILES	no	high	250k
ChemVAE	VAE	autoencoder	SMILES	yes	medium	250k
MolCycleGan	GAN	direct flow	latent vector	yes	medium	250k
ORGAN	GAN	RL	SMILES	yes	high	1million
ORGANIC	GAN	RL	SMILES	yes	high	250k
SSVAE	VAE	autoencoder	SMILES	yes	medium	310k
CDN	VAE	autoencoder	SMILES	no	low	250k
CVAE	VAE	autoencoder	SMILES	yes	medium	500k

Table 3.1: Qualitative comparison of the algorithms

To compare methods that were employed for the generation of molecules, several key characteristics of the machine learning algorithms have been examined: architecture, learning technique, molecular representations used, whether it can target desired property, computational resources needed to run it, and size of the training dataset. Please refer to the Table 3.1 for an overview.

3.6 Discussion

3.6.1 Performance of generation on demand

As was mentioned in the system overview section, the CDN was used as an algorithm for the generation of molecules on demand. To improve the proportion of correctly generated molecules the SMILES character parser have been substituted to a SMILES grammar parser. The original method of converting SMILES strings to number vectors involved assigning a number to each character of the SMILES string. For example, the atom H would be codified as 23, atom S as 24 and atom Si as a combination of two numbers 24 and 25 that should be placed consecutively.

Thus, if number 25 would appear as standalone in the resulting vector, the whole vector would be discarded because the corresponding string and associated molecule would be invalid. To eliminate such cases SMILES grammar parser have been used to break the SMILES string into morphemes, i.e. atoms and supporting elements, like stereoisomers. While the grammar parser does not eliminate syntactic errors it helps with standalone atom parts.

Utilizing the uniform training dataset for every generation request mainly resulted in a production of completely irrelevant molecules. However, when the application of case specific training datasets described in the previous section have been introduced the reliability of generation on demand featured has greatly improved.

During testing, it was observed that all the inputs for the generation on demand could be roughly divided into three categories: small molecules representing common structural motifs widely found in more complex molecules, medium-sized

molecules that are not so widespread as subunits for other molecules, and large complex molecules that cannot be identified as substructures of any molecules from ZINC, ChEMBL or cheML.io. Owing to these differences, inputs from each of the above categories might require their own approach for assembling the training datasets. For example, the similarity-based training dataset for small molecules could be readily assembled from any database. However, due to the small size of the input molecule, the resulting training dataset might include molecules that are rather different from the initial one in the sense that they would not contain it as a substructure. Thus, adding a substructure-based training dataset and blending it with a similarity-based training dataset generally led to a more balanced outcome for the generation requests featuring small and medium-sized molecules as inputs. On the other hand, for large and complex molecules that can not be found as substructures of other molecules, the only option is to use a similarity-based training dataset. Therefore, a 3-stage process for assembling the training datasets have been designed. It accounts for the above mentioned variations and provides optimal results for any type of input structure without the need for manual categorizing.

3.6.2 Analysis across methods

As can be seen in Figure 3-1, the aim was not to create a uniform number of molecules per method when implementing the studied algorithms. The main reason is due to the fact that some of the algorithms are more suitable for the generation of the bulk of molecules while others are more convenient for the targeted generation of specific molecules. For example, CVAE can be regarded as a specialized algorithm for the generation of molecules displaying specific properties while CDN is designed to generate similar yet diverse molecules when compared to a particular prototype. Thus, both CVAE and CDN were deployed by us for the generation of only a relatively small set of molecules ranging from several hundred to several thousand. On the other hand, considering its focus on structural similarity, CDN appeared to be the most suitable method for incorporating into the generation on demand feature.

While the majority of generation algorithms shows a rather diverse output, when

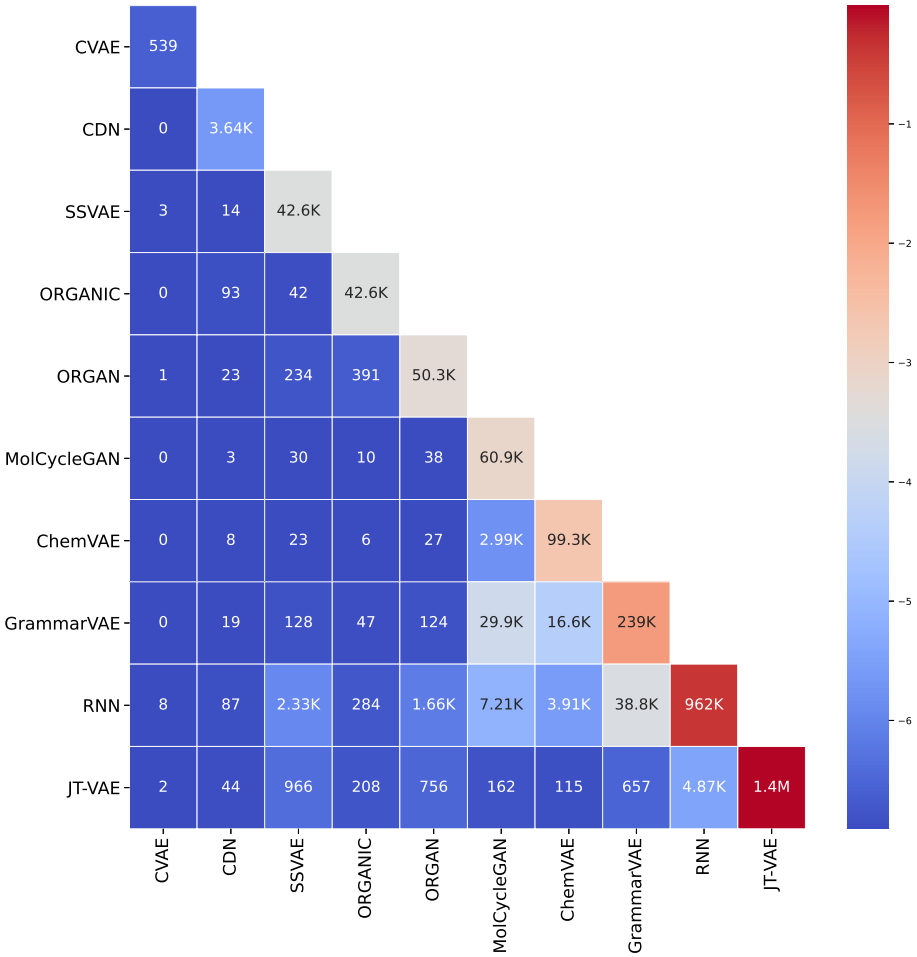


Figure 3-1: The diagonal of the matrix illustrates total number of molecules generated by each method. Intersections below the diagonal show number of same molecules that were generated by both methods.

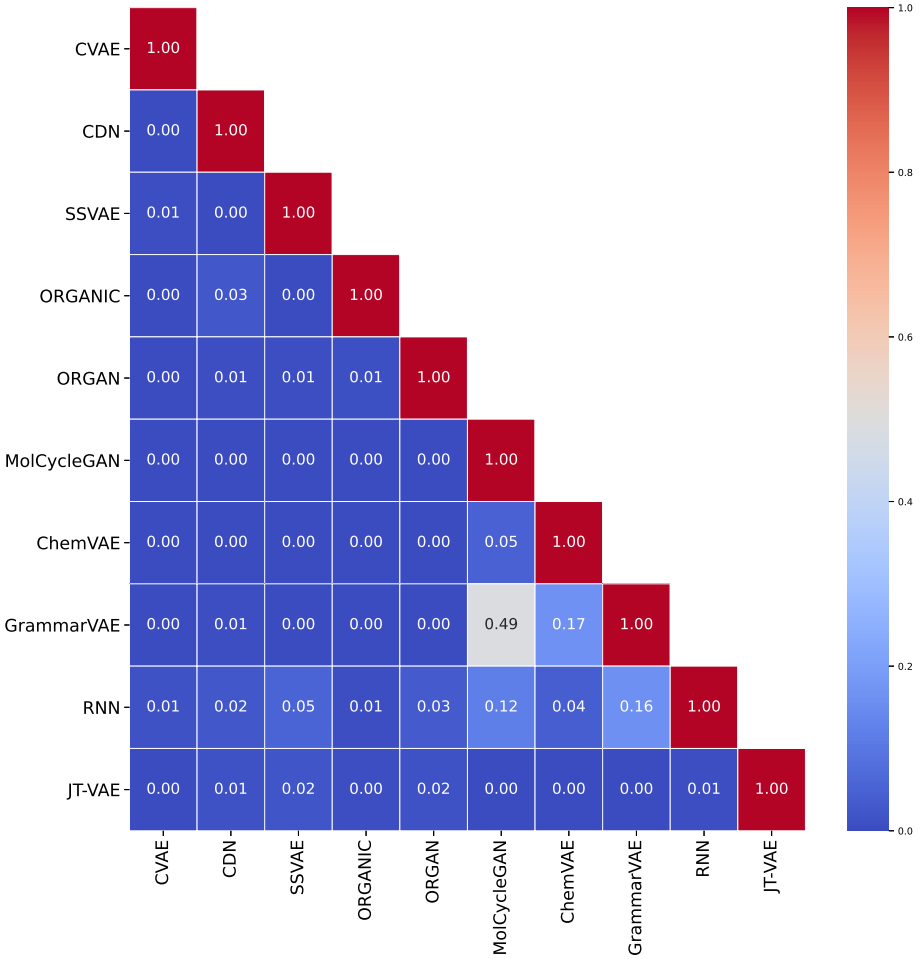


Figure 3-2: Each entry shows the proportion of shared molecules between each method.

compared to the output of other algorithms (see Figure 3-2), 49.07% of molecules generated by MolCycleGAN were also generated by GrammarVAE. This indicates

that these methods may have a similar latent space despite that MolCycleGAN uses direct graph representation of the molecule while GrammarVAE uses the context-free grammar of the SMILES representation.

3.6.3 Moses benchmark comparison

To juxtapose the methods that have been employed, the MOSES benchmark framework was used [34]. It is a benchmark that encompasses several metrics that assess the generated molecules in order to characterize a given method. One of the important metrics is *novelty*. It is a proportion of generated molecules that are not seen in the initial database, i.e. the training set. *Filters* is a metric that measures the proportion of molecules that are passed through the custom medicinal chemistry filters (MCFs) and PAINS filters [4]. These filters were hand-picked to exclude molecules with specific undesired properties, such as reactivity and chelation. $IntDiv_1$ and $IntDiv_2$ assess the internal diversity of the generated molecules and their values range between 0 and 1. A low score indicates that the generated molecules are limited in the variety of scaffolds and a high score corresponds to a higher inner diversity of the molecules.

$IntDiv_1$ and $IntDiv_2$ can be calulated as follows

$$IntDiv_p(G) = 1 - \sqrt[p]{\frac{1}{|G|^2} \sum_{m_1, m_2 \in G} T(m_1, m_2)^p},$$

where G stands for the set of molecules, T stands for Tanimoto distance, m_1 and m_2 stand for any pair of molecules in the set G and $p \in 1, 2$.

Model	**# of molecules**	**IntDiv$_1$**	**IntDiv$_2$**	**Filters**	**Novelty**
JT-VAE	1399265	0.861	0.856	0.733	1
RNN	962247	0.847	0.837	0.813	0.953
GrammarVAE	239262	0.871	0.865	0.598	0.196
ChemVAE	99344	0.879	0.874	0.589	1
MolCycleGan	60856	0.869	0.862	0.607	0.42
ORGAN	50268	0.86	0.852	0.71	0.902
ORGANIC	42610	0.855	0.842	0.621	0.999
SSVAE	42606	0.84	0.832	0.89	0.971
CDN	3639	0.886	0.878	0.620	0.997
CVAE	539	0.786	0.767	0.538	1

Table 3.2: Comparison of methods by means of the MOSES benchmark

As can be seen from Table 3.2, aside from GrammarVae and MolCycleGAN the algorithms have shown they are able to produce a large portion of novel, never-seen-before molecules. The internal diversity score also shows high values across all 10 algorithms.

Chapter 4

Transmol: Repurposing a language model for molecular generation

4.1 Introduction

After the completion of the previous project and analysis of the machine learning algorithms it became apparent that there is a gap in the literature. To my knowledge even though there are algorithms borrowed from the natural language processing, like CDN, no machine learning algorithm for *de novo* molecule generation has used attention mechanisms, especially state-of-the-art Transformer architecture. So this chapter is a natural continuation from the previous one. While Chapter 3 was discussing and using previously developed algorithms, this chapter demonstrates an ambition for the development of the independent method.

The goal is to develop an algorithm that allows the generation of a diverse focused libraries utilizing one, or two seed molecules which guide the generation of *de novo* molecules. The approach outperforms state-of-the-art generative machine learning frameworks in some core MOSES metrics, a benchmark introduced for the comparison of generative algorithms [34]: internal diversity (IntDiv_1 and InDiv_2). The resulting algorithm is incorporated into the cheml.io [53] website and can be utilized for the generation of molecules on demand. One or two seed molecules can be defined by the used and a focused library is generated.

4.2 Dataset

For this chapter, I have used the MOSES benchmark along with the dataset it provides. It consists of three datasets: training, testing, and testing scaffolds, containing 1.6M, 176k, and 176k respectively.

The first dataset was used to train the model. The model learns to interpolate between each molecule and constructs a latent space. The latent space acts as a proxy distribution for molecules, therefore it is possible to sample new molecules from it.

The testing dataset consists of molecules that are not present in the training dataset. It is used to assess how effectively the model is generalizable: whether the architecture of the model can be applied to other datasets.

The scaffold testing dataset consists of scaffolds that are not present in the training and testing datasets. Scaffolds are small fragments of molecules that can describe a set of compounds, where it is present. The scaffold testing is used to check if the model can generate new scaffolds, unique molecular features, or whether the model just reuses the parts of the previously seen molecules to generate new ones.

4.2.1 Molecular representation

In this chapter I have used SMILES strings as a molecular representations of choice. Please see Section 2.1.2 for the detailed description.

4.2.2 Data augmentation

To improve the validity of the algorithm I have used data augmentation through SMILES enumeration as was used in work of Arús-Pous et al. [3]. A molecule can be mapped to its unique canonical SMILES string, however non-unique SMILES strings can also be produced depending on the starting point where the algorithm will begin its translation. Such data augmentation has been previously reported to improve the generalization of the latent space (increase the diversity of the output molecules) [3].

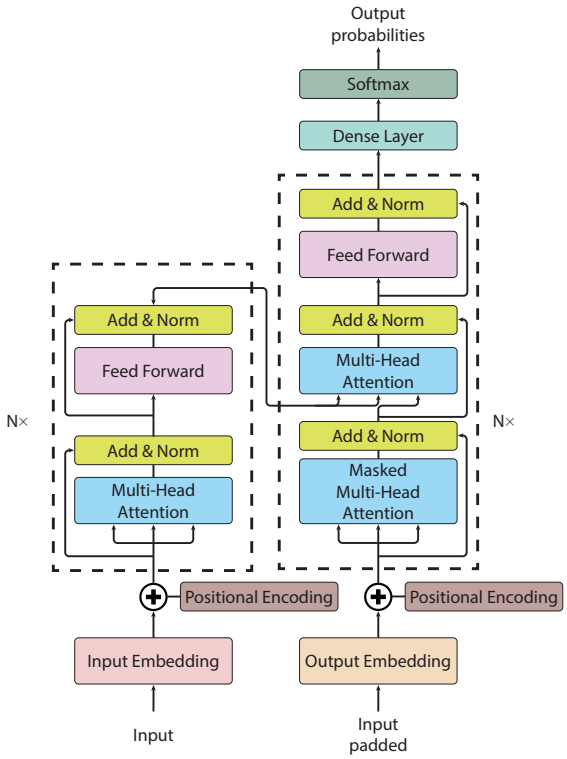


Figure 4-1: A vanilla transformer architecture

4.3 Method

For this work I have employed a vanilla transformer model from the work by Vaswani et al. [46]. A vanilla transformer consists of two parts: encoder and decoder. An encoder (see left dashed block of Figure 4-1) maps input to the latent representation z. A decoder (see right dashed block of Figure 4-1), accepts z as an input and produces one symbol at a time. The model is auto-regressive, i.e to produce a new symbol it requires the previous output as an additional input.

The notable attribute of this architecture is the use of attention mechanisms throughout the whole model. While models before transformers have been using attention only as an auxiliary layer, having some kind of recurrent neural networks (RNN) like gated recurrent unit (GRU) or long short-term memory (LSTM), or convolutional neural network (CNN), the transformer consists primarily of attention layers.

The attention mechanism can be looked at as function of query Q, key K and value

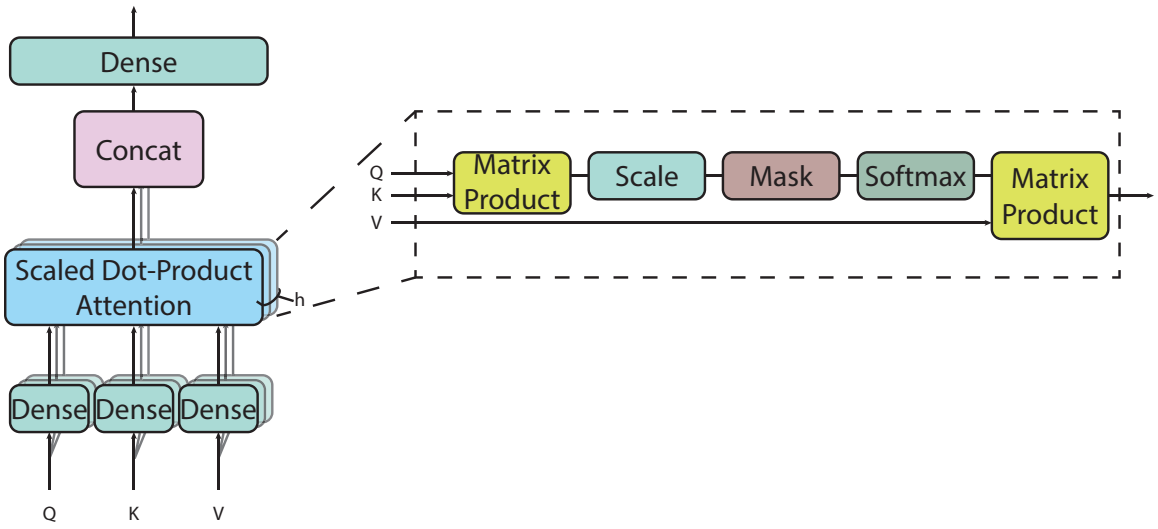


Figure 4-2: Multi-head attention layer

V, where the output is a matrix product of Q, K, V using the following function:

$$\text{Scaled dot-product Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{4.1}$$

It is used to identify the relevant parts of the input in respect to the input, self-attention. It allows to disregard less important parts of the query and filter noise. The most important part is that attention mechanisms are differentiable, hence can be learned from data.See Equation 4.1 for the desciption of the scaled dot-product attention layer. The multi-head attention layer consists of h instances of scaled dot-product attention layers that are then concatenated and passed to the dense layer.

The multi-head attention layer consists of h instances of scaled dot-product attention layers that are then concatenated and passed to the dense layer. See Figure 4-2 for the detailed depiction.

Parameters of the original setup have been used, such as number of stacked encoder and decoder layers $N = 6$, all sublayers produce output of $d_{\text{model}} = 512$, with dimensionality of inner feed-forward layer being d_{ff}, number of attention heads $h = 6$, and dropout $d = 0.1$.

4.3.1 Sampling from the latent space

To sample a molecule from the model, a seed SMILES string is needed to provide a context for the decoder. Then the decoding process is started by supplying a special starting symbol. After that, the decoder provides an output and a first symbol is generated. To get the next symbol the previous characters are provided to the decoder. The decoding process stops when the decoder either outputs a special terminal symbol or exceeds the maximum length. There are several techniques available that specify how the output of the decoder is converted to the SMILES character such as a simple greedy search or a beam search.

Greedy search

As the decoder provides output probabilities the naive approach would be to use a greedy algorithm and pick the symbol with the highest probability. However, it is not optimal as picking the most probable symbol at each step does not guarantee that the final resulting string would have the highest conditional probability. Moreover, unless stochastic sampling is used (when probability vector is used as a basis for the distribution and then sampled), the result of the greedy search is deterministic and corresponds to the "reconstruction accuracy" based on our training procedure.

Beam search

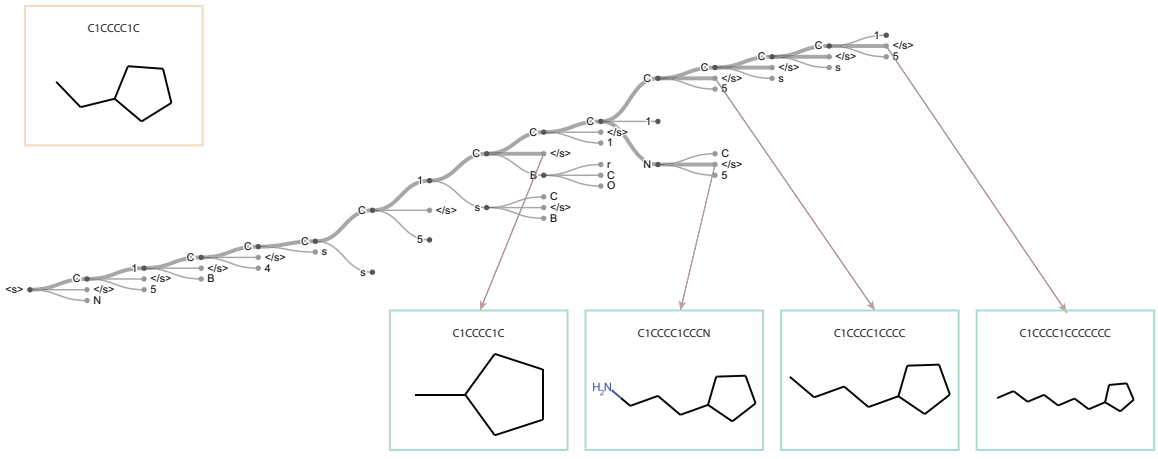


Figure 4-3: Overview of the beam search with a beam width of N=3

To improve upon the greedy search a beam search has been proposed. The beam search is an improved greedy search. While the greedy search picks only one symbol at a time, the beam search picks N most probable. Figure 4-3 illustrates the beam search with beam width $N = 3$; the stroke width indicates the probability. To guide the selection process of beam search I have used the following reward function:

$$\frac{\sum_{char \in vocab} P(s|\text{previous output})}{(1 + |\text{previous output}|)^{\alpha}}$$

where *char* is a possible symbol for the beam search to pick, *vocab* is a set of all possible characters, the *previous output* is an ordered list of symbols picked by beam search prior to picking current one, α is a parameter of beam search that regulates the length of the string, low α discourages long strings, high α encourages.

4.3.2 Injecting variability into model

To explore the molecules that are located near the seed molecule in the latent space, I have used two techniques that allow to sample from the seed cluster: addition of Gaussian noise to z and the use of temperature.

Gaussian noise

To increase the variability of the model I are adding the Gaussian noise with a mean μ and standard deviation σ to the latent vector z before it is fed to the decoder.

Temperature

Another technique to improve the variability is to apply temperature to the output vector right before applying the softmax function. Temperature T is a value from 0 to ∞. As $T \to \infty$ all characters have the same probability of being the next symbol. For $T \to 0$ the most probable symbol has a higher probability of being selected. The resulting smoothed distribution increases the variability of sampling. Figure 4-4 demonstrates how the application of temperature smoothes the original distribution.

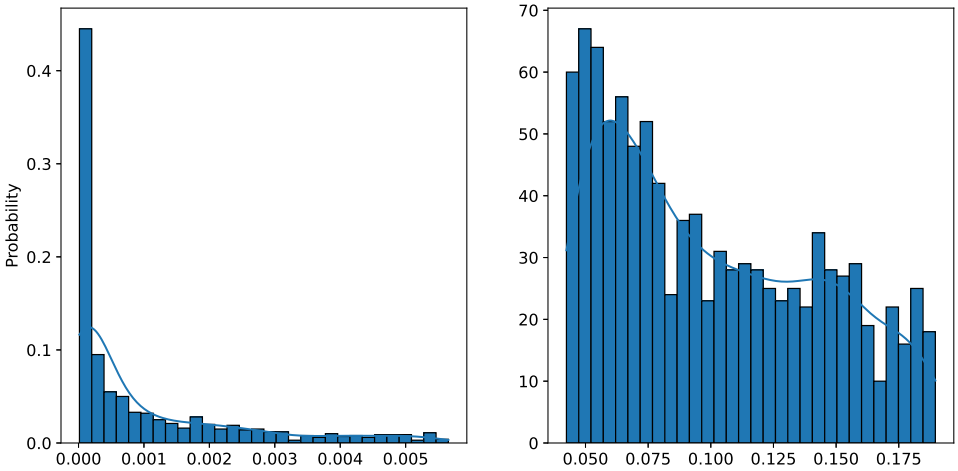


Figure 4-4: Impact of temperature on distribution

4.4 Results and Discussion

In this section, I describe major results that were obtained during the experiments. It starts with the generation of a focused library with a single seed molecule which is followed by the description of the generation of a focused library using two seed molecule. See Figure 4-5 for the graphical overview of the process.

4.4.1 Creating focused library with seed molecules

In this subsection, I discuss the optimization procedure for the sampling hyperparameters as well as present the results of the MOSES benchmark [34] in relation to our method and previous ones.

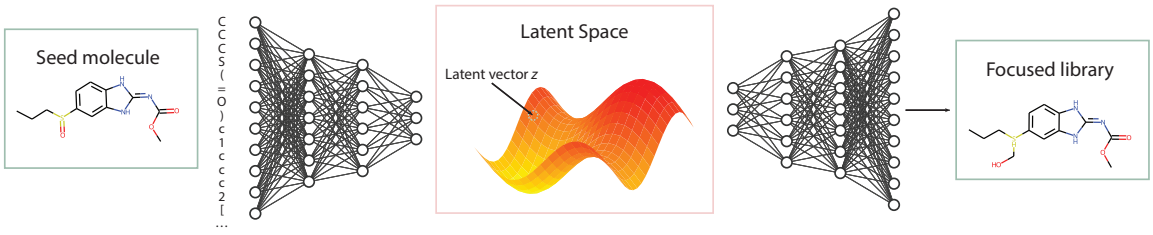


Figure 4-5: The general pipeline of the sampling process for one seed molecule

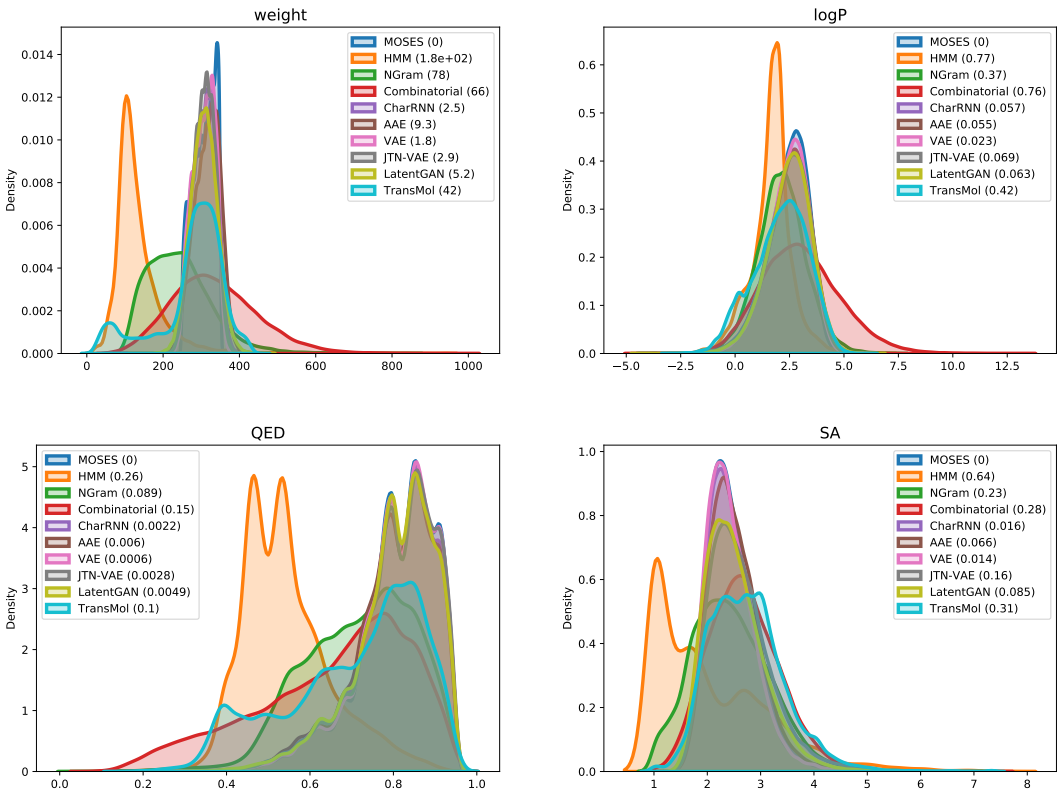


Figure 4-6: Plots of Wasserstein-1 distance between distributions of molecules in the generated and test sets

MOSES baselines

MOSES provides several implemented methods to compare our results to. They can be roughly divided into two categories: neural and non-neural. Neural methods use artificial neural networks to learn the distribution of the training set. Character-level recurrent neural network (CharRNN), Variation Autoencoder (VAE), Adversarial Autoencoder (AAE), Junction Tree VAE (JT-VAE), and Latent Vector Based Generative Adversarial Network (LatentGAN). Non-neural baselines include the n-gram generative model (NGram), the hidden Markov model (HMM), and a combinatorial generator. Non-neural baselines are conceptually simpler than neural ones. NGram model collects the frequency of the n-grams in the training dataset and uses the resulting distribution to sample new strings. For instance, during counting of 2-gram, the model will inspect individual SMILES strings and record the statistics. For string "C1CCC1C" the following statistics will be gathered C1: 2, CC:2, 1C:2. Later it will be normalized and used for sampling. HMM uses the Baum-Welch algorithm for the distribution learning. The combinatorial generator uses BRICS fragments of the training dataset. To sample it randomly connects several fragments.

Moses *metrics*

Several metrics are provided by the MOSES benchmark. Uniqueness shows the proportion of generated molecules that are within the training dataset. Validity describes the proportion of generated molecules that are chemically sound, as checked by RDKit [35]. Internal diversity measures whether the model samples from the same region of chemical space, producing molecules that are valid and unique but differ in a single atom; hence, are useless. Filters measures the proportion of generated set that passes a number of medical filters. Since the training set contains only molecules that pass through the filters, it is an implicit constraint imposed on the algorithm.

Fragment similarity (Frag) measures the similarity of BRICS fragments distribution contained in reference and generated sets. If the value is 1, then all fragments from the reference set are present in the generated one. If the value is 0, then there are

Model	Valid (↑)	Unique@1k (↑)	Unique@10k (↑)	IntDiv (↑)	IntDiv2 (↑)	Filters (↑)	Novelty (↑)
Train	*1*	*1*	*1*	*0.8567*	*0.8508*	*1*	*1*
HMM	0.076±0.0322	0.623±0.1224	0.5671±0.1424	0.8466±0.0403	0.8104±0.0507	0.9024±0.0489	**0.9994±0.001**
NGram	0.2376±0.0025	0.974±0.0108	0.9217±0.0019	0.8738±0.0002	0.8644±0.0002	0.9582±0.001	0.9694±0.001
Combinatorial	**1.0±0.0**	0.9983±0.0015	0.9909±0.0009	0.8732±0.0002	0.8666±0.0002	0.9557±0.0018	0.9878±0.0008
CharRNN	0.9748±0.0264	**1.0±0.0**	0.9994±0.0003	0.8562±0.0005	0.8503±0.0005	0.9943±0.0034	0.8419±0.0509
AAE	0.9368±0.0341	**1.0±0.0**	0.9973±0.002	0.8557±0.0031	0.8499±0.003	0.996±0.0006	0.7931±0.0285
VAE	0.9767±0.0012	**1.0±0.0**	0.9984±0.0005	0.8558±0.0004	0.8498±0.0004	**0.997±0.0002**	0.6949±0.0069
JTN-VAE	**1.0±0.0**	**1.0±0.0**	**0.9996±0.0003**	0.8551±0.0034	0.8493±0.0035	0.976±0.0016	0.9143±0.0058
LatentGAN	0.8966±0.0029	**1.0±0.0**	0.9968±0.0002	0.8565±0.0007	0.8505±0.0006	0.9735±0.0006	0.9498±0.0006
Transmol	0.0694±0.0004	0.9360±0.0036	0.9043±0.0036	**0.8819±0.0003**	**0.8708±0.0002**	0.8437±0.0015	0.9815±0.0004

Table 4.1: Performance metrics for baseline models: fraction of valid molecules, fraction of unique molecules from 1,000 and 10,000 molecules, internal diversity, fraction of molecules passing filters (MCF, PAINS, ring sizes, charge, atom types), and novelty. Reported (mean ± std) over three independent model initializations.

no overlapping fragments between generated and reference sets. Scaffold similarity (Scaff) is similar to the Frag, but instead of BRICS fragments Bemis–Murcko scaffolds are used for comparison. The range of this metric is similar to Frag. Similarity to the nearest neighbor (SNN) is a mean Tanimoto distance between a molecule in the reference set and its closest neighbor from the generated set. One of the possible interpretations of this metric is precision; if the value is low, it means that the algorithm generates molecules that are distant from the molecules in the reference set. The limits of this metric are [0,1]. Fréchet ChemNet Distance (FCD) is a metric that is supposed to correlate with internal diversity and uniqueness. It is computed using the penultimate layer of the ChemNet, a deep neural network that is trained for the prediction of biological activities. All four aforementioned metrics have also been compared to the scaffold test dataset.

According to Table 4.1, Transmol has demonstrated the greatest internal diversity

Model	FCD (↓)		SNN (↑)		Frag (↑)		Scaf (↑)	
	Test	TestSF	Test	TestSF	Test	TestSF	Test	TestSF
Train	*0.008*	*0.4755*	*0.6419*	*0.5859*	*1*	*0.9986*	*0.9907*	*0*
HMM	24.4661±2.5251	25.4312±2.5599	0.3876±0.0107	0.3795±0.0107	0.5754±0.1224	0.5681±0.1218	0.2065±0.0481	0.049±0.018
NGram	5.5069±0.1027	6.2306±0.0966	0.5209±0.001	0.4997±0.0005	0.9846±0.0012	0.9815±0.0012	0.5302±0.0163	0.0977±0.0142
Combinatorial	4.2375±0.037	4.5113±0.0274	0.4514±0.0003	0.4388±0.0002	0.9912±0.0004	0.9904±0.0003	0.4445±0.0056	0.0865±0.0027
CharRNN	**0.0732±0.0247**	**0.5204±0.0379**	0.6015±0.0206	0.5649±0.0142	**0.9998±0.0002**	0.9983±0.0003	0.9242±0.0058	**0.1101±0.0081**
AAE	0.5555±0.2033	1.0572±0.2375	0.6081±0.0043	0.5677±0.0045	0.991±0.0051	0.9905±0.0039	0.9022±0.0375	0.0789±0.009
VAE	0.099±0.0125	0.567±0.0338	**0.6257±0.0005**	**0.5783±0.0008**	0.9994±0.0001	**0.9984±0.0003**	**0.9386±0.0021**	0.0588±0.0095
JTN-VAE	0.3954±0.0234	0.9382±0.0531	0.5477±0.0076	0.5194±0.007	0.9965±0.0003	0.9947±0.0002	0.8964±0.0039	0.1009±0.0105
LatentGAN	0.2968±0.0087	0.8281±0.0117	0.5371±0.0004	0.5132±0.0002	0.9986±0.0004	0.9972±0.0007	0.8867±0.0009	0.1072±0.0098
Transmol	4.3729±0.0466	5.3308±0.0428	0.6160±0.0005	0.4614±0.0007	0.9564±0.0009	0.9496±0.0009	0.7394±0.0009	0.0183±0.0065

Table 4.2: Performance metrics for baseline models: Fréchet ChemNet Distance (FCD), Similarity to a nearest neighbor (SNN), Fragment similarity (Frag), and Scaffold similarity (Scaff); Reported (mean ± std) over three independent model initializations. Results for random test set (Test) and scaffold split test set (TestSF)

(IntDiv_1 and IntDiv_1) across all baselines. It can be also observed that among neural algorithms Transmol demonstrates the greatest proportion of novel molecules, that are not present in the training dataset. One of the important observations is that Transmol's internal diversity score exceeds the training dataset. This observation might indicate the ability of Transmol to generalize well on the previously not seen data.

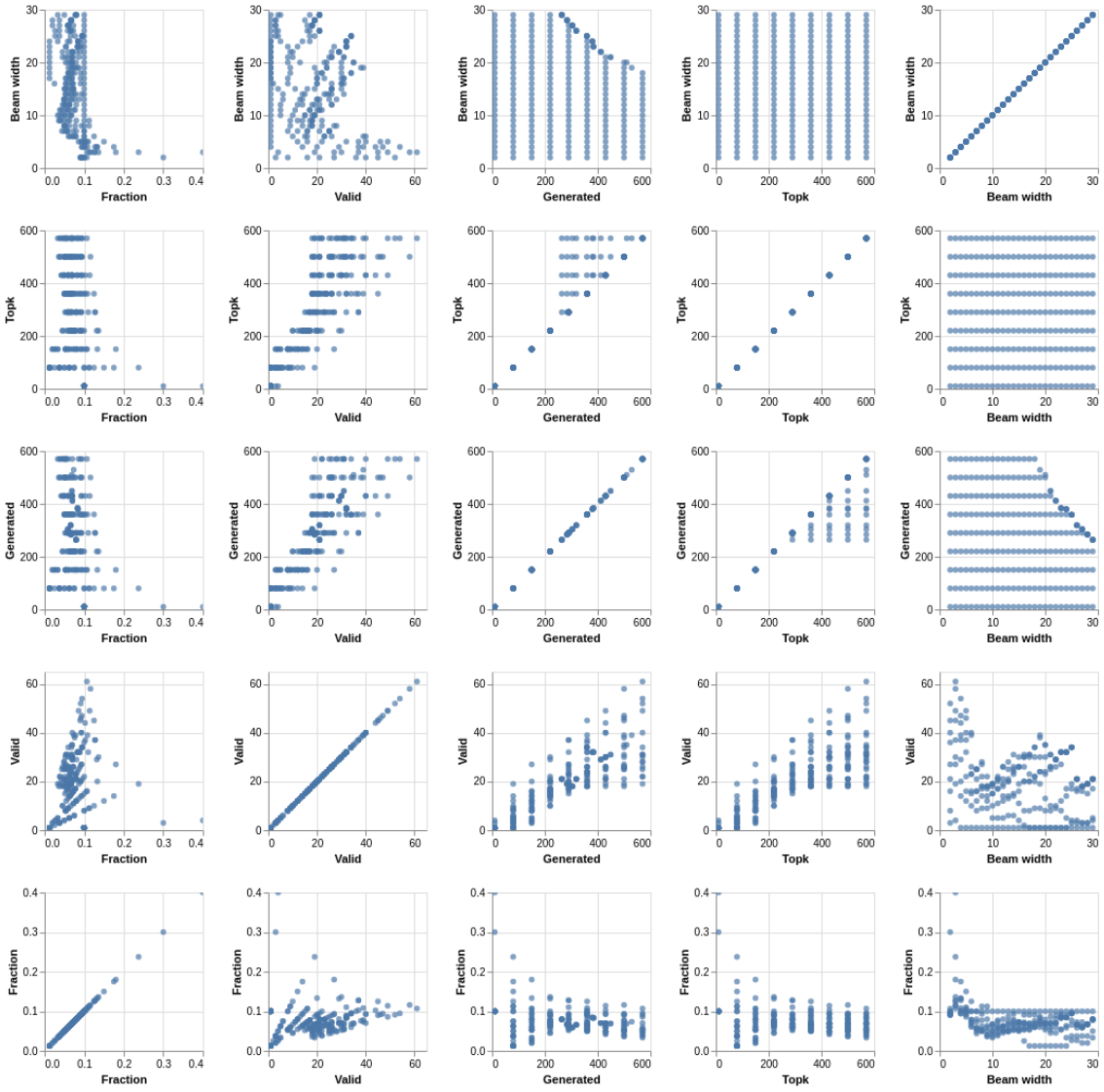


Figure 4-7: The scatter plot of the grid search with the following parameters: beam width, number of generation requests, actual number of generated smiles, number of valid molecules, and fraction of valid molecules

Table 4.2 shows more sophisticated metrics that require comparison of the two

molecular sets, reference and generated one: Fréchet ChemNet Distance (FCD), Similarity to the nearest neighbor (SNN), Fragment similarity (Frag), and Scaffold similarity (Scaff). In this table, it can be observed that Transmol has a relatively high FCD score compared with neural methods and a comparable or lower score in relation to non-neural algorithms. It is a surprising result, considering the high scores of the Transmol in internal diversity and quite high scores in uniqueness. Another observation is that Transmol demonstrates the superiority in SNN/Test and Scaf/Test compared with non-neural baselines and is comparable to other neural algorithms. In SNN/Test Transmol is a top-2 algorithm. Another thing to note is the TestSF column. The original authors of the benchmark recommend a comparison of the TestSF columns when the goal is to generate molecules with scaffolds that are not present in the training set. However, the comparison with caution as the test scaffold set is not all-encompassing. It does not contain all possible scaffolds that are absent in the training dataset. Taking into consideration that metrics in Table 4.2 compute overlaps in the two sets, the TestSF part of the metrics should be taken with a grain of salt.

Figure 4-6 demonstrates distribution plots of the baselines and Transmol output compared to the test set. The distribution plots are similar to the histograms, but instead of showing discrete bins, the distribution plot smoothes observations using Gaussian kernel. The distribution plots compare 4 molecular properties: molecular weight (MW), octanol-water partition coefficient (logP), quantitative estimation of drug-likeness (QED), and synthetic accessibility score (SA). To quantify the distance between the test set and the generated set the Wasserstein-1 distance was used (value in brackets). The results show that the Transmol has a matching SA score, or better than the original distribution while having a higher variance in other metrics. It shows that the Transmol is not as close to the testing set distribution as other neural algorithms, implying a better diversity, but it is not as far from it as some simpler, combinatorial baselines.

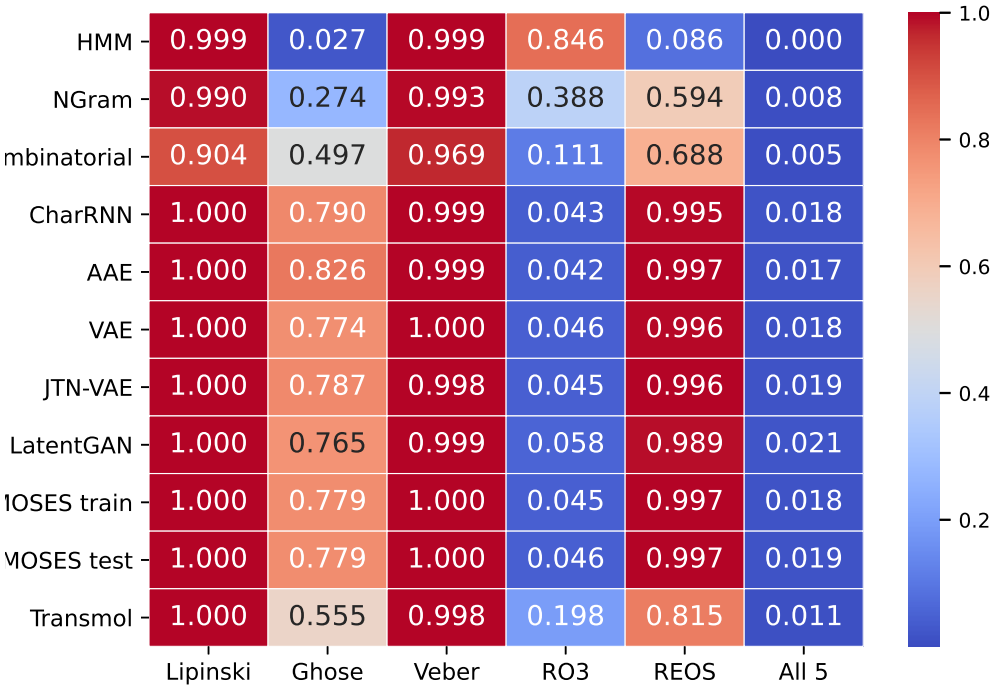


Figure 4-8: Proportions of molecule that satisfy 5 rules of thumb

4.4.2 Filters

To provide more comparison I also have filtered molecules by using heuristics that are used in drug development. These filters were empirically devised using a database of drug suitable compounds. For this comparison, Ihave used the Lipinski rule of 5 [29], Ghose [14], Veber [47], rule of 3 [10], and REOS [48] filters. Figure 4-8 demonstrates proportions of molecules that satisfy each rule.

As can be seen in the Figure 4-8 among the neural baselines Transmol has the highest proportion of molecules that satisfy the rule of 3. In addition, among non-neural algorithms, Transmol has the highest proportion of molecules that satisfy the Ghose filter and REOS.

4.4.3 Adjusting beam search

To find the optimal parameters for the beam search, a grid search has been conducted. A random molecule from the test set has been selected for the testing. Figure 4-7 shows the relation between 3 parameters (Beam width, Topk(the desired number of generated molecules), Generated (actual number of generated molecules)), and 2 metrics (number of valid molecules and the fraction valid molecules in the generated set). The chart demonstrates a dependency between beam width and topk(number of generated molecules). In addition, for small numbers of topk, the fraction of valid molecules is high. Similarly, Figure B-1 describes the relation between temperature, Gaussian standard deviation, fraction of valid molecules and internal diversity IntDiv_1 and IntDiv_2.

4.4.4 Exploring chemical space using two seed molecules

After sampling of the model using a single seed a natural question to ask if the approach could be extended further. In this section, I discuss the generation of a focused library using two seed molecules. Figure 4-9 demonstrates an overview of the sampling. Using the encoder network of the Transmol model we encode two molecules, getting their latent representation. Then, they are averaged to get a latent vector z_{12} that is located in the middle between latent vector z_1 and z_2. After that, the decoder is sampled using vector z_{12}. To increase the chance of sampling from the populated latent space I enumerate SMILES of seed molecules and construct pairs. This twist increases the chances of sampling the middle point that contains valid SMILES strings.

Since no known benchmark involves multiseed sampling in this subsection I describe a procedure of adjusting the weights. Figure 4-10 illustrates molecular sampling from the latent distribution using two seed molecules. The resulting generated library demonstrates a diversity of structural features that would be unattainable through simple fragment substitution.

Adjusting weights

After verification of the encoder, the natural way of proceeding would be trying to generate intermediate representations of the molecules, and decode molecules that resemble both seed one and seed two. See Figure 4-9 for the illustration of intermediate representation in the context of latent space. Additional comparisons of weight distributions and the sampling reward of beam search (α) can be found in Appendix A, Tables A.1 to A.6.

4.4.5 Integration with cheMl.io

Currently anyone with the access to the internet is able to use the Transmol via the cheml.io [53] website, see Figure 4-11. The form on the website allows generation with one or two seed molecules as well as specification of some other parameters like the weight of specific molecule and an α parameter, which regulates the length of the output molecule. The results are sent to the specified email address.

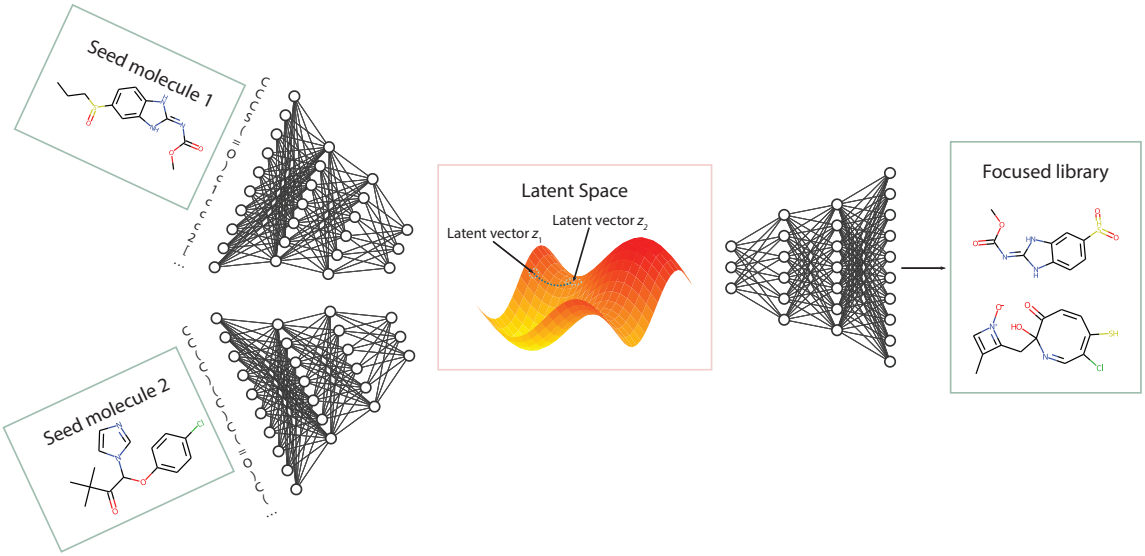


Figure 4-9: The general pipeline of the sampling process for two seed molecule

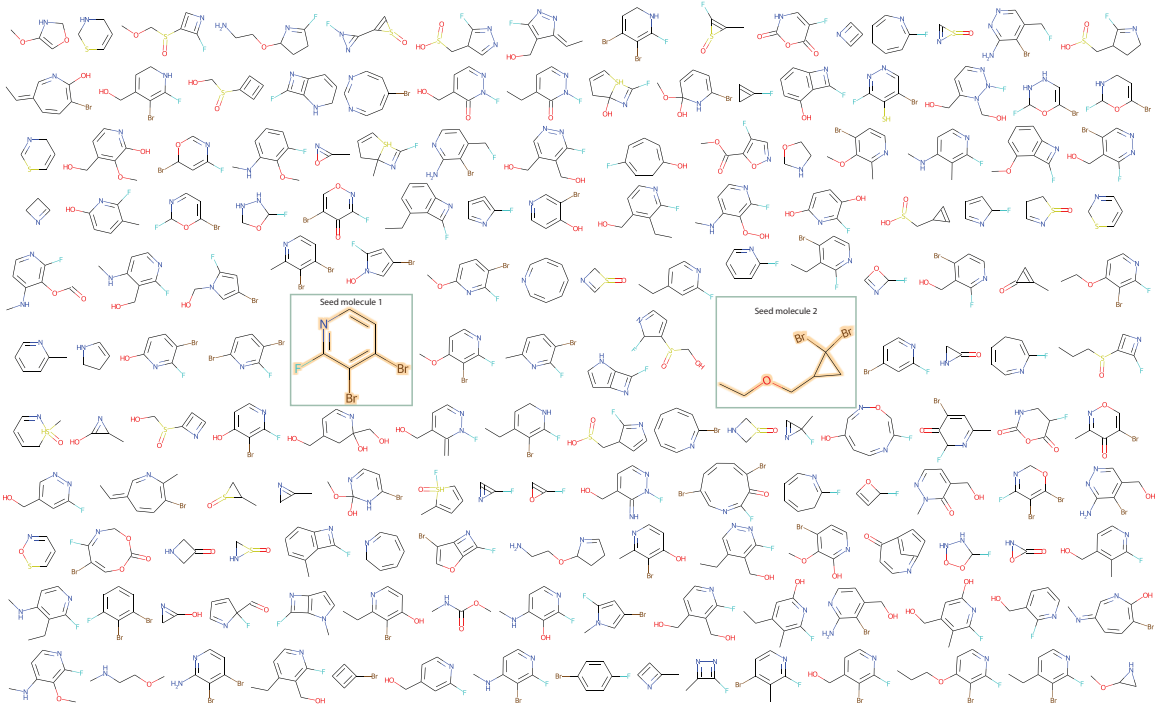


Figure 4-10: Example sampling of two molecules

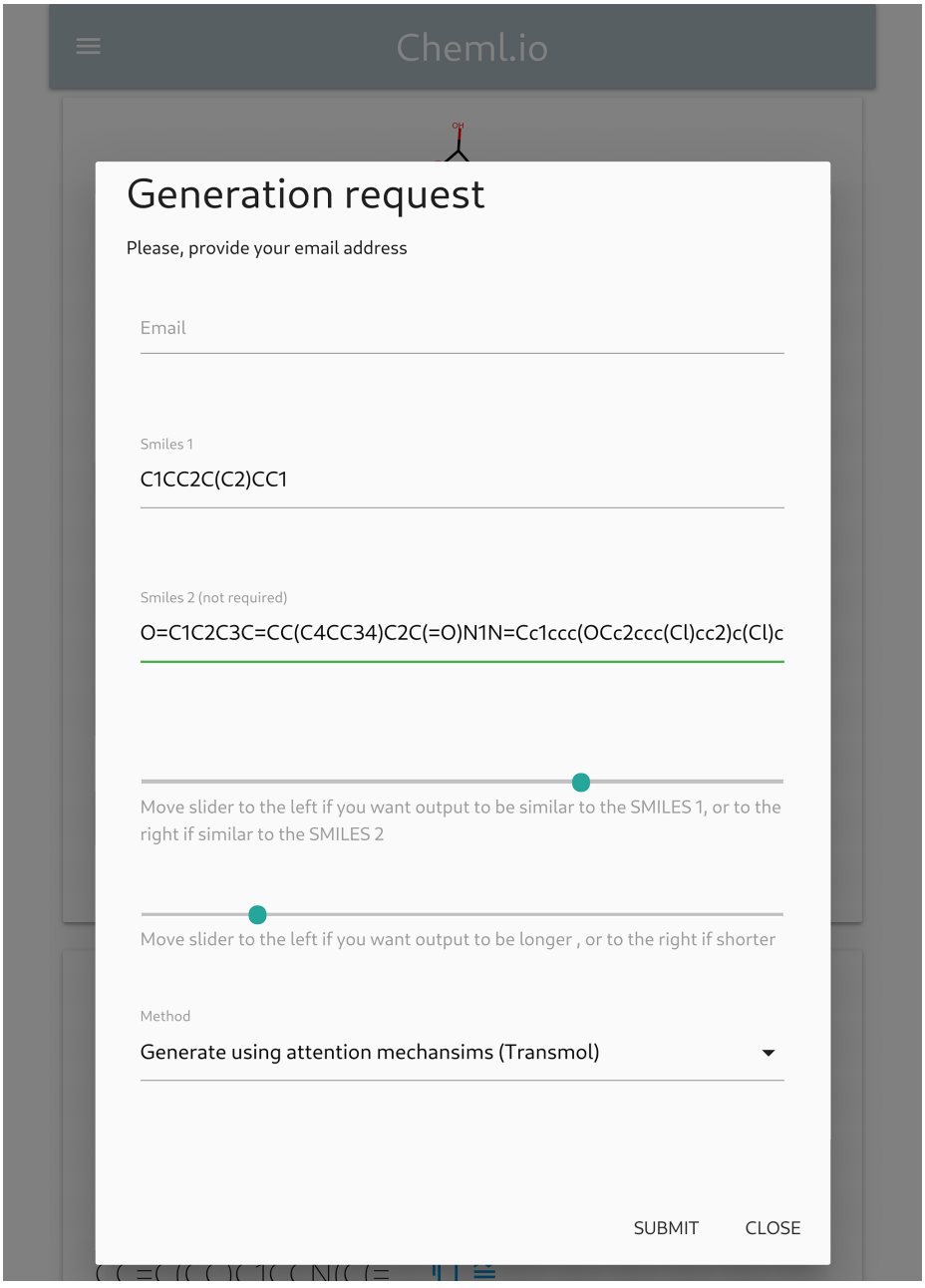


Figure 4-11: Screenshot of the generation request form on cheml.io

Chapter 5

Conclusions

In summary, an accessible way of using machine learning algorithms was provided as the website cheML.io. Recent machine learning algorithms were investigated. The resulting generated molecules have been inserted into a database that allows substructure and similarity searches. When results are suboptimal, new molecules could be generated on the user's demand. A recent deep learning framework has been successfully applied to the task of molecular generation using attention mechanisms. We have benchmarked the resulting Transmol method utilizing the MOSES benchmark. The results demonstrate a number of the advantages when using this attention-based methodology in comparison with earlier approaches. In addition, such model architecture allows the generation of new focused molecular libraries using two seeds. I believe that the proposed work will be helpful in the field of drug design, where novel molecules are used for the creation of life-saving drugs.

Appendix A

Tables

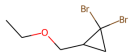
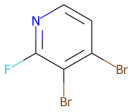
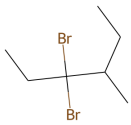
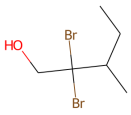
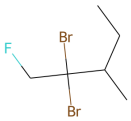
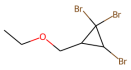
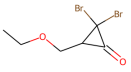
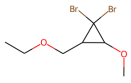
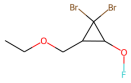
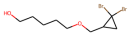
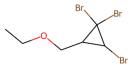
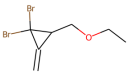
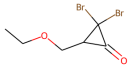
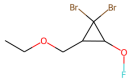
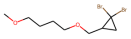
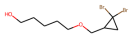
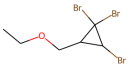
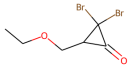
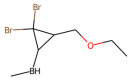
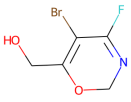
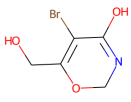
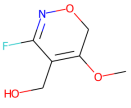
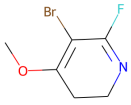
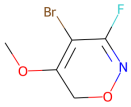
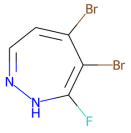
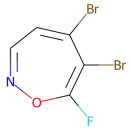
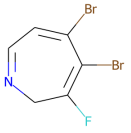
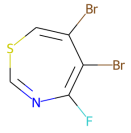
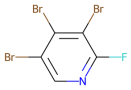
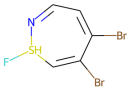
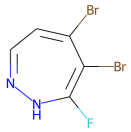
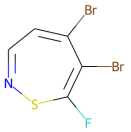
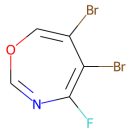
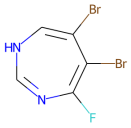
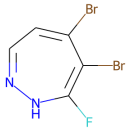
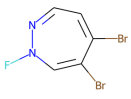
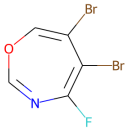
Seed molecule 1 				Seed molecule 2 			
No	Weight 1	Weight 2	Molecule 1	Molecule 2	Molecule 3	Molecule 4	Molecule 5
1	0.55	0.45					
2	0.65	0.35					
3	0.75	0.25					
4	0.85	0.15					
5	0.45	0.55					
6	0.35	0.65					
7	0.25	0.75					
8	0.15	0.85					

Table A.1: Comparison of weights for the first pair of molecules

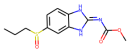
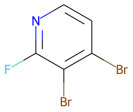
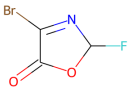
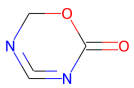
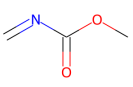
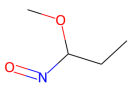
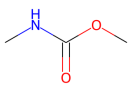
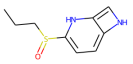
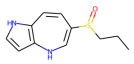
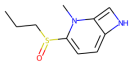
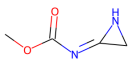
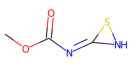
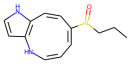
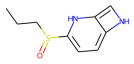
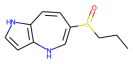
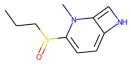
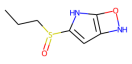
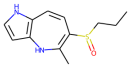
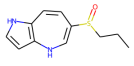
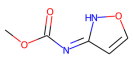
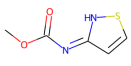
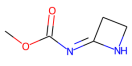
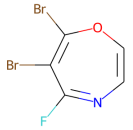
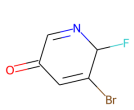
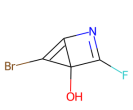
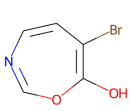
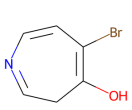
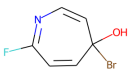
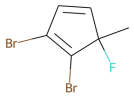
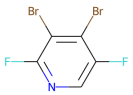
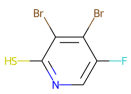
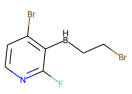
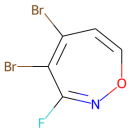
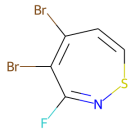
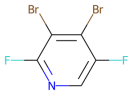
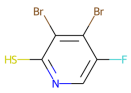
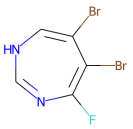
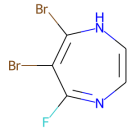
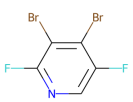
Seed molecule 1				Seed molecule 2			
							
No	Weight 1	Weight 2	Molecule 1	Molecule 2	Molecule 3	Molecule 4	Molecule 5
1	0.55	0.45					
2	0.65	0.35					
3	0.75	0.25					
4	0.85	0.15					
5	0.45	0.55					
6	0.35	0.65					
7	0.25	0.75					
8	0.15	0.85					

Table A.2: Comparison of weights for the second pair of molecules

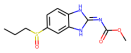
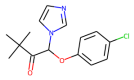
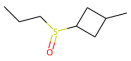
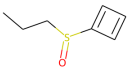
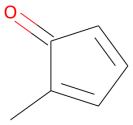
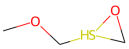
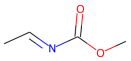
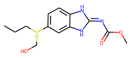
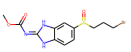
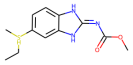
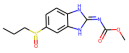
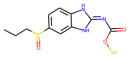
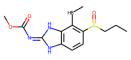
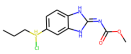
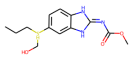
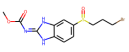
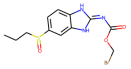
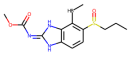
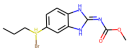
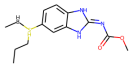
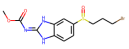
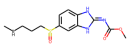
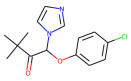
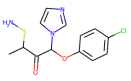
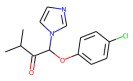
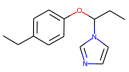
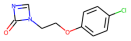
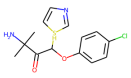
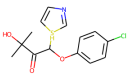
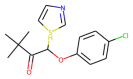
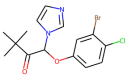
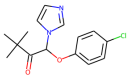
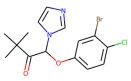
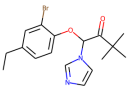
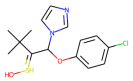
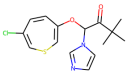
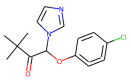
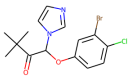
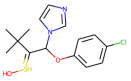
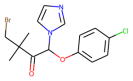
Seed molecule 1				Seed molecule 2			
							
No	Weight 1	Weight 2	Molecule 1	Molecule 2	Molecule 3	Molecule 4	Molecule 5
1	0.55	0.45					
2	0.65	0.35					
3	0.75	0.25					
4	0.85	0.15					
5	0.45	0.55					
6	0.35	0.65					
7	0.25	0.75					
8	0.15	0.85					

Table A.3: Comparison of weights for the third pair of molecules

Seed molecule 1										Seed molecule 2													
No	Sampling reward	Length				Tanimoto similarity 1				Tanimoto similarity 2				Tanimoto similarity (1+2)				Molecule 1	Molecule 2	Molecule 3	Molecule 4	Molecule 5
		min	max	mean	median	min	max	mean	median	min	max	mean	median	min	max	mean	median					
1	4	13	20	16.2	16	0.223	0.486	0.376	0.369	0.189	0.521	0.275	0.256	0.46	0.885	0.651	0.645					
2	3.7	13	20	16.2	16	0.223	0.486	0.371	0.367	0.189	0.521	0.265	0.25	0.46	0.885	0.635	0.638					
3	3.4	13	20	15.9	16	0.223	0.486	0.368	0.369	0.168	0.521	0.262	0.242	0.46	0.916	0.63	0.633					
4	3.1	11	21	16.2	16	0.223	0.486	0.374	0.378	0.168	0.487	0.255	0.242	0.46	0.916	0.629	0.637					
5	2.8	10	21	15.9	16	0.223	1	0.39	0.377	0.166	0.487	0.262	0.246	0.46	1.3	0.652	0.635					
6	2.5	10	23	16.2	16	0.223	1	0.389	0.37	0.166	0.521	0.262	0.246	0.46	1.3	0.651	0.628					
7	2.2	10	22	16	16	0.223	1	0.386	0.368	0.162	0.521	0.26	0.241	0.46	1.3	0.645	0.618					
8	1.9	5	22	15.6	16	0.175	1	0.381	0.372	0.159	0.521	0.265	0.241	0.334	1.3	0.646	0.617					
9	1.6	1	22	14.7	15	0	1	0.37	0.365	0	0.521	0.271	0.248	0	1.3	0.641	0.616					
10	1.35	1	19	14	15	0	1	0.364	0.358	0	0.702	0.28	0.255	0	1.3	0.644	0.63					
11	1.3	1	20	13.6	15	0	1	0.363	0.363	0	0.516	0.288	0.272	0	1.3	0.651	0.628					
12	1.2	1	19	13.3	14	0	1	0.356	0.356	0	0.702	0.284	0.265	0	1.3	0.64	0.633					
13	1.05	1	19	12.6	13	0	1	0.349	0.354	0	0.516	0.289	0.283	0	1.3	0.639	0.639					
14	0.9	1	19	11.7	12	0	1	0.339	0.353	0	0.516	0.291	0.294	0	1.3	0.629	0.642					
15	0.75	1	19	10.9	11	0	0.893	0.328	0.352	0	0.524	0.289	0.29	0	1.18	0.617	0.634					
16	0.6	1	19	9.53	10	0	0.893	0.31	0.333	0	0.516	0.278	0.287	0	1.18	0.588	0.633					
17	0.45	1	17	8.23	8	0	0.706	0.287	0.314	0	0.516	0.268	0.294	0	1.12	0.555	0.627					
18	0.3	1	16	7.39	7	0	0.694	0.263	0.305	0	0.516	0.253	0.286	0	1.12	0.516	0.596					
19	0.15	1	14	6.45	6	0	0.694	0.236	0.246	0	0.516	0.233	0.258	0	1.12	0.469	0.502					
20	0	1	14	6.03	6	0	0.492	0.222	0.22	0	0.516	0.223	0.238	0	1.01	0.445	0.468					

Table A.4: Comparison of sampling reward for the first pair of molecules

Seed molecule 1 

Seed molecule 2 

No	Sampling reward	Length min				Tanimoto mol1 min				Tanimoto mol2 min				Tanimoto sum min				Molecule 1	Molecule 2	Molecule 3	Molecule 4	Molecule 5
		min	max	mean	median	min	max	mean	median	min	max	mean	median	min	max	mean	median					
1	2.5	13	23	17.3	17	0.258	0.438	0.346	0.348	0.18	1	0.355	0.268	0.438	1.32	0.701	0.657					
2	2.1	13	26	15.7	15	0.229	0.5	0.352	0.327	0.176	1	0.343	0.317	0.476	1.32	0.695	0.67					
3	1.7	10	22	15.4	15	0.228	0.5	0.352	0.329	0.176	1	0.346	0.318	0.476	1.32	0.698	0.645					
4	1.35	1	26	14.8	15	0	0.498	0.348	0.374	0	1	0.321	0.288	0	1.32	0.669	0.646					
5	1.3	1	22	14.3	15	0	0.5	0.344	0.334	0	1	0.332	0.31	0	1.32	0.676	0.677					
6	1.2	1	20	14.4	15	0	0.498	0.341	0.373	0	1	0.321	0.305	0	1.32	0.662	0.652					
7	1.05	1	20	12.9	15	0	0.5	0.313	0.337	0	1	0.302	0.301	0	1.32	0.615	0.641					
8	0.9	1	20	12	14	0	0.5	0.305	0.329	0	1	0.295	0.293	0	1.32	0.599	0.634					
9	0.75	1	20	10.7	13	0	0.5	0.284	0.322	0	1	0.276	0.288	0	1.32	0.559	0.613					
10	0.6	1	20	9.92	10	0	0.5	0.278	0.322	0	1	0.273	0.291	0	1.32	0.551	0.625					
11	0.45	1	18	9.1	8	0	0.498	0.271	0.323	0	0.702	0.261	0.291	0	1.08	0.533	0.625					
12	0.3	1	18	8.4	8	0	0.498	0.257	0.312	0	0.702	0.249	0.281	0	1.08	0.506	0.598					
13	0.15	1	19	7.26	7	0	0.516	0.236	0.259	0	0.524	0.227	0.249	0	1.04	0.463	0.541					
14	0	1	17	6.71	7	0	0.547	0.226	0.203	0	0.556	0.22	0.206	0	1.1	0.446	0.409					

Table A.5: Comparison of sampling reward for the second pair of molecules

Seed molecule 1																		Seed molecule 2				
No	Sampling reward	Length min				Tanimoto mol1 min				Tanimoto mol2 min				Tanimoto sum min				Molecule 1	Molecule 2	Molecule 3	Molecule 4	Molecule 5
		min	max	mean	median	min	max	mean	median	min	max	mean	median	min	max	mean	median					
1	6.7	36	36	36	36	0.36	0.36	0.36	0.36	0.426	0.426	0.426	0.426	0.786	0.786	0.786	0.786					
2	6.1	36	36	36	36	0.36	0.36	0.36	0.36	0.426	0.426	0.426	0.426	0.786	0.786	0.786	0.786					
3	5.5	36	36	36	36	0.36	0.36	0.36	0.36	0.426	0.426	0.426	0.426	0.786	0.786	0.786	0.786					
4	4.9	36	36	36	36	0.36	0.36	0.36	0.36	0.426	0.426	0.426	0.426	0.786	0.786	0.786	0.786					
5	4.3	36	36	36	36	0.36	0.36	0.36	0.36	0.426	0.426	0.426	0.426	0.786	0.786	0.786	0.786					
6	3.7	36	36	36	36	0.36	0.36	0.36	0.36	0.426	0.426	0.426	0.426	0.786	0.786	0.786	0.786					
7	3.1	36	36	36	36	0.36	0.36	0.36	0.36	0.426	0.426	0.426	0.426	0.786	0.786	0.786	0.786					
8	2.5	36	36	36	36	0.36	0.372	0.365	0.362	0.426	0.5	0.453	0.435	0.786	0.862	0.818	0.807					
9	1.9	30	44	35.6	36	0.349	0.42	0.373	0.367	0.409	0.501	0.447	0.433	0.784	0.862	0.82	0.82					
10	1.35	1	32	9.55	9	0	0.531	0.322	0.362	0	0.531	0.322	0.363	0	1.06	0.644	0.724					
11	1.3	1	20	9.13	9	0	0.406	0.234	0.234	0	0.406	0.233	0.234	0	0.812	0.467	0.469					
12	1.2	1	32	9.07	7.5	0	0.547	0.297	0.348	0	0.547	0.297	0.352	0	1.09	0.594	0.704					
13	1.05	1	32	7.76	7	0	0.547	0.264	0.258	0	0.547	0.264	0.258	0	1.09	0.528	0.516					
14	0.9	1	13	6.33	6	0	0.547	0.244	0.219	0	0.547	0.244	0.219	0	1.09	0.487	0.438					
15	0.75	1	13	6.34	6	0	0.547	0.239	0.203	0	0.547	0.239	0.203	0	1.09	0.477	0.406					
16	0.6	1	13	6.25	6	0	0.547	0.228	0.203	0	0.547	0.228	0.203	0	1.09	0.456	0.406					
17	0.45	1	13	5.98	6	0	0.547	0.216	0.188	0	0.547	0.216	0.188	0	1.09	0.432	0.375					
18	0.3	1	13	5.73	6	0	0.547	0.204	0.156	0	0.547	0.204	0.156	0	1.09	0.408	0.312					
19	0.15	1	13	5.6	6	0	0.547	0.196	0.156	0	0.547	0.196	0.156	0	1.09	0.393	0.312					
20	0	1	12	5.48	6	0	0.547	0.188	0.156	0	0.547	0.188	0.156	0	1.09	0.377	0.312					

Table A.6: Comparison of sampling reward for the third pair of molecules

Appendix B

Figures

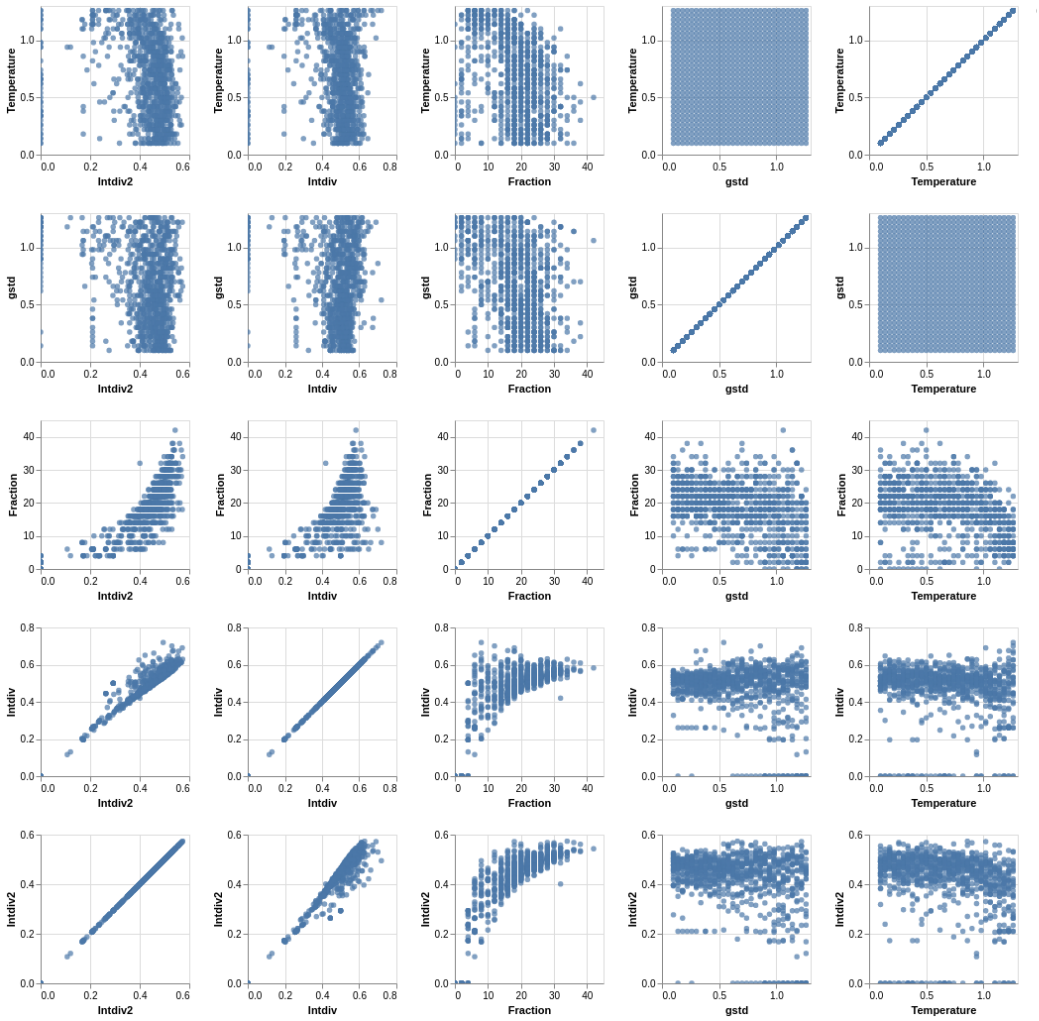


Figure B-1: The scatter plot of the grid search with the following parameters: temperature, standard deviation of the Gaussian noise, fraction of valid molecules, $IntDiv_1$, and $IntDiv_2$

Bibliography

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, USA, 2016. USENIX Association.

[2] Samar Y. Al-nami, Enas Aljuhani, Ismail Althagafi, Hana M. Abumelha, Tahani M. Bawazeer, Amerah M. Al-Solimy, Zehba A. Al-Ahmed, Fatimah Al-Zahrani, and Nashwa El-Metwaly. Synthesis and Characterization for New Nanometer Cu(II) Complexes, Conformational Study and Molecular Docking Approach Compatible with Promising in Vitro Screening. *Arabian Journal for Science and Engineering*, 46(1):365–382, January 2021.

[3] Josep Arús-Pous, Simon Viet Johansson, Oleksii Prykhodko, Esben Jannik Bjerrum, Christian Tyrchan, Jean-Louis Reymond, Hongming Chen, and Ola Engkvist. Randomized SMILES strings improve the quality of molecular generative models. *Journal of Cheminformatics*, 11(1):71, December 2019.

[4] Jonathan B. Baell and Georgina A. Holloway. New Substructure Filters for Removal of Pan Assay Interference Compounds (PAINS) from Screening Libraries and for Their Exclusion in Bioassays. *J. Med. Chem.*, 53(7):2719, April 2010.

[5] Alexandru T. Balaban. Solved and Unsolved Problems in Chemical Graph Theory. In *Annals of Discrete Mathematics*, volume 55, pages 109–126. Elsevier, 1993.

[6] A. T. Brint and P. Willett. Upperbound procedures for the identification of similar three-dimensional chemical structures. *Journal of Computer-Aided Molecular Design*, 2(4):311–320, January 1989.

[7] Hans-Joachim Böhm. The computer program LUDI: A new method for the de novo design of enzyme inhibitors. *Journal of Computer-Aided Molecular Design*, 6(1):61–78, February 1992.

[8] Andrea Cadeddu, Elizabeth K. Wylie, Janusz Jurczak, Matthew Wampler-Doty, and Bartosz A. Grzybowski. Organic Chemistry as a Language and the Im-

plications of Chemical Linguistics for Structural and Retrosynthetic Analyses. *Angewandte Chemie International Edition*, 53(31):8108–8112, July 2014.

[9] A. CAUCHY. Methode generale pour la resolution des systemes d'equations simultanees. *C.R. Acad. Sci. Paris*, 25:536–538, 1847.

[10] Miles Congreve, Robin Carr, Chris Murray, and Harren Jhoti. A 'Rule of Three' for fragment-based lead discovery? *Drug Discovery Today*, 8(19):876–877, October 2003.

[11] Sourav Das, Sharat Sarmah, Sona Lyndem, and Atanu Singha Roy. An investigation into the identification of potential inhibitors of SARS-CoV-2 main protease using molecular docking study. *Journal of Biomolecular Structure and Dynamics*, pages 1–11, May 2020.

[12] Michael B. Eisen, Don C. Wiley, Martin Karplus, and Roderick E. Hubbard. HOOK: A program for finding novel molecular architectures that satisfy the chemical and steric requirements of a macromolecule binding site. *Proteins: Structure, Function, and Genetics*, 19(3):199–221, July 1994.

[13] Daniel C. Elton, Zois Boukouvalas, Mark D. Fuge, and Peter W. Chung. Deep learning for molecular design—a review of the state of the art. *Mol. Syst. Des. Eng.*, 4(4):828, 2019.

[14] Arup K. Ghose, Vellarkad N. Viswanadhan, and John J. Wendoloski. A Knowledge-Based Approach in Designing Combinatorial or Medicinal Chemistry Libraries for Drug Discovery. 1. A Qualitative and Quantitative Characterization of Known Drug Databases. *Journal of Combinatorial Chemistry*, 1(1):55–68, January 1999.

[15] Valerie J. Gillet, William Newell, Paulina Mata, Glenn Myatt, Sandor Sike, Zsolt Zsoldos, and A. Peter Johnson. SPROUT: Recent developments in the de novo design of molecules. *Journal of Chemical Information and Modeling*, 34(1):207–217, January 1994.

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

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts, 2016.

[18] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q.

Weinberger, editors, *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672. MIT Press, Cambridge, MA, USA, 2014.

[19] Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. arxiv, February 2018, preprint, arXiv:1705.10843v3. `https://arxiv.org/abs/1705.10843v3`.

[20] Shahar Harel and Kira Radinsky. Accelerating prototype-based drug discovery using conditional diversity networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18*, page 331, London, United Kingdom, 2018. ACM Press.

[21] Paul C. D. Hawkins. Conformation Generation: The State of the Art. *Journal of Chemical Information and Modeling*, 57(8):1747–1756, August 2017.

[22] A. G. Ivakhnenko. Polynomial Theory of Complex Systems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-1(4):364–378, October 1971.

[23] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, page 2323, Stockholmsmässan, Stockholm Sweden, July 2018. PMLR.

[24] Seokho Kang and Kyunghyun Cho. Conditional molecular design with deep generative models. *J. Chem. Inf. Model.*, 59(1):43, January 2019.

[25] Kentaro Kawai, Naoya Nagata, and Yoshimasa Takahashi. De Novo Design of Drug-Like Molecules by a Fragment-Based Molecular Evolutionary Approach. *Journal of Chemical Information and Modeling*, 54(1):49–56, January 2014.

[26] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, page 1945. JMLR. org, 2017.

[27] Y. Le Cun and Françoise Fogelman-Soulié. Modèles connexionnistes de l'apprentissage. *Intellectica*, 1:114, 1987.

[28] Jaechang Lim, Seongok Ryu, Jin Woo Kim, and Woo Youn Kim. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *J. Cheminf.*, 10(1):31, December 2018.

[29] Christopher A. Lipinski, Franco Lombardo, Beryl W. Dominy, and Paul J. Feeney. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv. Drug. Deliver. Rev.*, 23(1):3, 1997.

[30] Łukasz Maziarka, Agnieszka Pocha, Jan Kaczmarczyk, Krzysztof Rataj, and Michał Warchoł. Mol-cycleGAN - a generative model for molecular optimization. In Igor V. Tetko, Věra Kůrková, Pavel Karpov, and Fabian Theis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2019: Workshop and Special Sessions*, page 810, Cham, 2019. Springer International Publishing.

[31] David Mendez, Anna Gaulton, A Patrícia Bento, Jon Chambers, Marleen De Veij, Eloy Félix, María Paula Magariños, Juan F Mosquera, Prudence Mutowo, Michał Nowotka, María Gordillo-Marañón, Fiona Hunter, Laura Junco, Grace Mugumbate, Milagros Rodriguez-Lopez, Francis Atkinson, Nicolas Bosc, Chris J Radoux, Aldo Segura-Cabrera, Anne Hersey, and Andrew R Leach. ChEMBL: Towards direct deposition of bioassay data. *Nucleic Acids Res.*, 47(D1):D930, November 2018.

[32] Nataraj S. Pagadala, Khajamohiddin Syed, and Jack Tuszynski. Software for molecular docking: a review. *Biophysical Reviews*, 9(2):91–102, April 2017.

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, page 8024. Curran Associates, Inc., Red Hook, NY, USA, 2019.

[34] Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Simon Johansson, Hongming Chen, Sergey Nikolenko, Alan Aspuru-Guzik, and Alex Zhavoronkov. Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. *Frontiers in Pharmacology*, 2020.

[35] RDKit: Open-source cheminformatics. `http://www.rdkit.org`. (accessed September 2020).

[36] Jean-Louis Reymond. The chemical space project. *Accounts of Chemical Research*, 48(3):722–730, 3 2015.

[37] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *J. Chem. Inf. Model.*, 50(5):742, 2010.

[38] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[39] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986.

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

[41] Benjamin Sanchez-Lengeling, Carlos Outeiral, Gabriel L. Guimaraes, and Alan Aspuru-Guzik. Optimizing distributions over molecular space. An Objective-Reinforced Generative Adversarial Network for Inverse-design Chemistry (ORGANIC), 8 2017.

[42] Marwin H. S. Segler, Thierry Kogej, Christian Tyrchan, and Mark P. Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS Cent. Sci.*, 4(1):120, January 2018.

[43] Teague Sterling and John J. Irwin. ZINC 15 – ligand discovery for everyone. *J. Chem. Inf. Model.*, 55(11):2324, November 2015.

[44] Michael Stonebraker and Lawrence A. Rowe. The design of POSTGRES. *ACM SIGMOD Record*, 15(2):340–355, June 1986.

[45] John H. Van Drie, David Weininger, and Yvonne C. Martin. ALADDIN: An integrated tool for computer-assisted molecular design and pharmacophore recognition from geometric, steric, and substructure searching of three-dimensional molecular structures. *Journal of Computer-Aided Molecular Design*, 3(3):225–251, September 1989.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[47] Daniel F. Veber, Stephen R. Johnson, Hung-Yuan Cheng, Brian R. Smith, Keith W. Ward, and Kenneth D. Kopple. Molecular Properties That Influence the Oral Bioavailability of Drug Candidates. *Journal of Medicinal Chemistry*, 45(12):2615–2623, June 2002.

[48] W. Patrick Walters and Mark Namchuk. Designing screens: how to make your hits a hit. *Nature Reviews Drug Discovery*, 2(4):259–266, April 2003.

[49] David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Modeling*, 28(1):31–36, February 1988.

[50] Xiaolin Xia, Jianxing Hu, Yanxing Wang, Liangren Zhang, and Zhenming Liu. Graph-based generative models for de novo drug design. *Drug Discovery Today: Technologies*, 32-33:45–53, 2019. Artificial Intelligence.

[51] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 2852, San Francisco, California, USA, 2017. AAAI Press.

[52] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference On Computer Vision (ICCV)*, page 2242, 2017.

[53] Rustam Zhumagambetov, Daniyar Kazbek, Mansur Shakipov, Daulet Maksut, Vsevolod A. Peshkov, and Siamac Fazli. cheml.io: an online database of ml-generated molecules. *RSC Adv.*, 10:45189–45198, 2020.