Carl von Ossietzky
Universität
Oldenburg

Department für Informatik
Abteilung für Computational Intelligence

In Silico Design of Drug-Like Molecules with Evolutionary Algorithms and Transformers

Von der Fakultät für Informatik der Carl von Ossietzky Universität Oldenburg zur Erlangung des Grades und Titels eines

Doktor der Naturwissenschaften (Dr. rer. nat.)

angenommene Dissertation von

Tim Cofala

Erstgutachter: Prof. Dr. Oliver Kramer
Externer Zweitgutachter: Prof. Dr. Mike Preuss

Tag der Disputation: 4. April 2023

Zusammenfassung

Die Optimierung bekannter Moleküle und das Entdecken zuvor gänzlich unbekannter Substanzen ist ein Motor für den Fortschritt in den unterschiedlichsten Bereichen des alltäglichen Lebens. Molekulares Design treibt die Forschung nach neuen Medikamente voran, kann bei der Bindung von Treibhausgasen helfen oder die Effizienz der Stromerzeugung erhöhen. Allerdings ist der Raum potenziell möglicher Substanzen riesig, und neben den gewünschten Moleküleigenschaften müssen unterschiedlichste sekundäre Faktoren wie Synthetisierbarkeit und Stabilität der Moleküle berücksichtigt werden. In den letzten Jahren hat dies zu einem stetigen Anstieg der Forschung im Bereich des computergestützten Moleküldesigns geführt, bei dem zunehmend Techniken der künstlichen Intelligenz eingesetzt werden. Ein zentraler Aspekt der Forschung ist dabei dir Frage, in welcher Repräsentation die Moleküle gespeichert und verarbeitet werden. Da jede Repräsentation ihre eigenen Vor- und Nachteile aufweist, finden diverse Repräsentationen Anwendung: als einfache Zeichenketten, als Graphen aus Atomen und Bindungen, als drei-dimensionale Punktewolken oder molekulare Fingerabdrücke.

Das Ziel dieser Arbeit ist es, State-of-the-Art Methoden der künstlichen Intelligenz (KI) auf diese Repräsentationen zu adaptieren, um ihre Potenziale für die Generierung und Optimierung von Molekülen zu nutzen. Dabei legt die Arbeit besonderen Fokus auf die Anforderungen, die jede Repräsentation an die verwendeten Verfahren stellt und die Möglichkeiten, die sie eröffnet. Auf methodischer Seite werden dafür zum einen evolutionäre Algorithmen zur Optimierung von Molekülen verwendet. Darüber hinaus werden Transformer, eine Modellarchitektur aus der Sprachverarbeitung, auf die Generierung von Molekülen adaptiert. Die durchgeführten Experimente zeigen, dass sich ein auf String-basierten Repräsentationen arbeitender evolutionäre Algorithmen für die Optimierung von Molekülen anwenden lässt, was an dem Beispiel der Proteaseinhibition untersucht wird. Der Algorithmus kann Moleküle hinsichtlich ihrer Bindungsaffinität zu einem Zielprotein verbessern, allerdings müssen zusätzliche sekundäre Optimierungsziele berücksichtigt werden, um das Generieren realistischer Moleküle zu forcieren. Generative neuronale Modelle hingegen bieten die Möglichkeit auf Datenbanken bekannter Moleküle zu lernen und deren strukturellen Eigenschaften bei der Generierung neuer Moleküle zu berücksichtigen. Die folgenden Experimente demonstrieren, wie Transformer als solch generative Modelle für die Erzeugung von Molekülen im dreidimensionalen Raum eingesetzt werden können und wie dieser Ansatz auch größere molekulare Graphen unter Verwendung einer effizienteren Generierungsstrategie erzeugen kann. In einer abschließenden Studie zeigt sich, dass ein Zusammenspiel aus einem Transformer, der molekulare Fingerabdrücke erzeugt, und einem evolutionären Algorithmus zur Rekonstruktion dieser Fingerabdrücke, neue Möglichkeiten für die Molekülgenerierung eröffnet.

Zusammengenommen zeigen die Ergebnisse neue Methoden auf, wie KI-Verfahren für die Generierung von Molekülen in unterschiedlichen Repräsentationen genutzt werden können. Dabei sind Transformer ein wirksames Instrument für die Verarbeitung von Molekülen mit sequenziellen Repräsentationen und zur Erzeugung realistischer Moleküle. Evolutionäre Algorithmen können für die Optimierung von Molekülen und das Durchschreiten des molekularen

Suchraums eingesetzt werden. Eine Kombination der Verfahren bietet die Möglichkeit, diese Vorteile zu vereinen.

Abstract

Optimizing known molecules and discovering previously unknown substances is fuelling progress in various areas of everyday life. Molecular design drives research for new drugs that can help bind greenhouse gases or increase the efficiency of electricity generation. However, the space of potential substances is vast, and in addition to the desired molecular properties, a wide variety of secondary factors, such as the synthesizability and stability of the molecules, must be taken into account. In recent years, this has led to a steady increase in research on computational molecular design, with increasing use of artificial intelligence techniques. A central aspect of the research is the question in which representation the molecules are stored and processed. Since each has its advantages and disadvantages, diverse representations are finding application: as simple strings, as graphs of atoms and bonds, as three-dimensional point clouds, or as molecular fingerprints.

This work aims to adapt state-of-the-art artificial intelligence (AI) methods to these representations to exploit their potential for generating and optimizing molecules. In doing so, the work focuses on the requirements that each representation poses for the methods used and the possibilities that they open up. On the methodological side, evolutionary algorithms are used to optimize molecules. Furthermore, transformers, a neural network architecture from language processing, are adapted to the generation of molecules. The experiments show that an evolutionary algorithm working on string-based representations can be applied to the optimization of molecules, which is investigated using the example of protease inhibition. The algorithm can improve molecules in terms of their binding affinity to a target protein, but additional secondary optimization goals must be considered to force the generation of realistic molecules. On the other hand, generative neural models offer the possibility to learn from databases of known molecules and consider their structural properties when generating new molecules. The following experiments demonstrate how transformers can be used as a model for generating molecules in three-dimensional space and how this approach can also generate larger molecular graphs using a more efficient generation strategy. A final study shows that an interaction between a transformer that generates molecular fingerprints and an evolutionary algorithm to reconstruct these fingerprints opens up new possibilities for molecule generation.

In summary, the results reveal new methods for using AI techniques to generate molecules in different representations. In this context, transformers are an effective tool for processing molecules with sequential representations and generating realistic molecules. Evolutionary algorithms can be used for optimizing molecules and traversing the molecular search space. A combination of the methods offers the possibility to combine these advantages.

Contents

1 Introduction **1**
- 1.1 Overview of the Thesis . . . 7
- 1.2 Contributions . . . 9

I Foundations **11**

2 Core Concepts of AI **13**
- 2.1 Problem-solving and Optimization . . . 13
- 2.2 Learning . . . 14
- 2.3 Deep Generative Learning . . . 16

3 The Role of AI in Molecule Design **19**
- 3.1 Organic Chemistry . . . 19
- 3.2 Molecular Representations . . . 21
  - 3.2.1 Graphs . . . 21
  - 3.2.2 Strings . . . 22
  - 3.2.3 Spatial . . . 23
  - 3.2.4 Fingerprints . . . 24
- 3.3 Applications of AI in Molecule Design . . . 25
- 3.4 The Impact of De Novo Molecule Design . . . 30
- 3.5 Ethical Implications . . . 32

II Molecule Design **35**

4 Evolutionary Multi-Objective Design of SARS-CoV-2 Protease Inhibitor Candidates **37**
- 4.1 Genetic Algorithms . . . 37
- 4.2 Virus Protease Inhibition . . . 39
- 4.3 Related Work: Genetic Algorithms and Molecule Design . . . 40
- 4.4 Evolutionary Multi-Objective Molecule Search . . . 41
  - 4.4.1 Molecule Metrics . . . 41
  - 4.4.2 The SELFIES Representation . . . 43
  - 4.4.3 Genetic Operations . . . 44
  - 4.4.4 Fitness Evaluation . . . 45
  - 4.4.5 NSGA-II . . . 46
- 4.5 Experiments . . . 47
  - 4.5.1 Metric Development . . . 47
  - 4.5.2 Candidate Comparison . . . 50
- 4.6 Conclusion . . . 51

5 Spatial Generation of Molecules with Transformers **53**
5.1 Transformers . . . 54
5.1.1 Transformer Architecture . . . 54
5.1.2 Attention Mechanism . . . 56
5.1.3 Transformer Applications . . . 58
5.2 Related Work: Molecular Generation Models . . . 59
5.3 Transformers for Spatial Molecule Generation . . . 60
5.3.1 Data . . . 61
5.3.2 Architecture . . . 62
5.3.3 Training . . . 63
5.3.4 Sampling . . . 63
5.4 Experiments . . . 63
5.4.1 Generating Molecules from Scratch . . . 64
5.4.2 Completing Molecules . . . 67
5.5 Conclusion . . . 67

6 Transformers for Molecular Graph Generation **69**
6.1 Related Work: Graph Generative Models . . . 70
6.2 Transformers for Graph Generation . . . 71
6.2.1 Data Representation . . . 71
6.2.2 Architecture . . . 71
6.2.3 Training and Sampling . . . 72
6.3 Experiments . . . 73
6.4 Conclusion . . . 74

7 An Evolutionary Fragment-based Approach to Molecular Fingerprint Reconstruction **77**
7.1 Genetic Algorithms for Molecular Fingerprint Reconstruction . . . 79
7.1.1 Genetic Algorithm . . . 79
7.1.2 Representation and Molecule Data . . . 80
7.1.3 Genetic Operators . . . 82
7.2 Experiments . . . 83
7.2.1 Reconstruction of Molecules . . . 83
7.2.2 Fingerprint Transformer . . . 87
7.3 Conclusion . . . 90

III Conclusion and Future Work **93**

8 Conclusion **95**
8.1 Representations . . . 95
8.2 Genetic Algorithms and Transformers . . . 99
8.3 Future Work . . . 100

References **103**

List of Figures **117**

List of Tables **119**

1 Introduction

"The ability of AI to design molecules is not only a game changer for the pharmaceutical industry, but has the potential to revolutionize the entire field of medicine."

GPT-3[1]

Today, the search for new molecules and their synthesis is essential in industry and research. In many domains of our everyday life, progress is directly linked to the discovery of molecules with desired properties, e.g., in drug discovery, crop protection, and chemical biology [Bro+20a]. Furthermore, humanity is in constant need of new materials to address the major environmental challenges of our time, be it energy production and storage or greenhouse gas conversion [But+18]. Overall, the discovery of new molecules can bring tremendous societal and technological advances [Pol+20]. Traditionally, this design process is driven by experimentation and guided by the experience and intuition of experts [Che+20]. The role of computers in this process has consisted mainly of modeling and simulation. The underlying physical principles of chemistry are well understood, and consequently, computers can facilitate the search for new molecules by predicting their behavior from the fundamental laws of physics alone [But+18].

However, the emergence of artificial intelligence (AI) has the potential, as for many domains in science and engineering, to change the role of computers in chemistry [But+18]. For one, AI can facilitate material research by improving the prediction of molecular properties, predicting phase diagrams and crystal structures, and increasing the speed of material simulations [MKR16]. Moreover, AI can assist in discovering entirely new molecules tailored to the specific use case. The possibility of generating molecules directly for the task at hand carries an enormous societal and technological impact and has, e.g., applications in energy production and energy storage. The importance of AI in material design is further highlighted by the fact that data-driven approaches form a core concept of the U.S. Materials Genome Initiative [MKR16]. One of the most important potential applications for AI-guided molecule generation is healthcare. In health care, effective means of discovering new molecules bear immense potential, promising personalized and precise medicine and the potential of curing rare diseases [Pol+20]. Throughout this thesis, molecule generation will be addressed, with a particular focus on the construction of drug-like molecules.

Despite constant improvements and the high importance of the task, the discovery of new molecules remains a complicated matter. A significant reason is the enormous theoretical

[1] GPT-3 is a language model based on deep learning [Bro+20b].

space of molecules, which, e.g., has been estimated to be between 10^{23} to 10^{80} in size for pharmacologically-sensible molecules [Ram+14; Rud+12; Pol+20]. This enormous number of potential molecules, even when limited by molecular size, renders a complete experimental or computational exploration infeasible [Bro+20a]. In addition, developing new drugs and materials requires optimization of a wide range of parameters and consideration of additional constraints such as synthetic accessibility [Bro+20a]. Chemical engineers are thus faced with complex multi-objective optimization problems, and even to this day, the question of how to effectively navigate the search space of potential molecules without having to enumerate significant parts of it remains unanswered [Bro+19].

Identifying a suitable molecule candidate for the task at hand resembles searching for a needle in a haystack, so the methodologies applied are constantly evolving. So far, there have been three distinct paradigms of material discovery [Che+20]. Firstly, the conventional experimentally-driven, trial-and-error molecule design. This approach is guided by expert knowledge, experience, and intuition. The increasing computational power and advancements in simulation and modeling tools enabled the second stage of material design. Computer simulations have made it possible to model the molecular properties of a compound before it has been synthesized, opening up new possibilities in finding the suitable molecule [But+18]. Tools like molecular dynamics simulations or density functional theory made it possible to analyze numerous known and hypothetical molecules to predict their structure, and behavior [But+18]. These advancements have led to large databases of molecules and their properties and introduced high-throughput virtual screening as a routine technique for molecular search [Che+20]. Virtual screening is the process of searching huge compound databases *in silico* with the help of computational methods to predict desired properties, similarities, or binding affinities to a target structure [Baj02]. It complements high-throughput screening (HTS), one of the most prominent techniques in molecule design to date. In HTS, molecule libraries are experimentally tested for specific target properties utilizing robotics and autonomous detection techniques. Although HTS has advanced the drug discovery process in the past two decades, the estimated hit rates (finding a compound with the desired properties) are relatively low, ranging between 0 and 0.01% [Sch+20]. However, it is of essential importance for the success of a drug discovery project to select promising HTS hits as starting points for the subsequent steps in the design process [Sch+20]. Overall, both introduced paradigms of molecule discovery bear disadvantages, as they resemble Edisonian approaches [MKR16]. Relying solely on expert knowledge and intuition can result in the by-chance discovery and a loss of generality in addition to being extremely time-, labor-, and cost-consuming [Che+20]. Although automating this process with HTS has had a significant influence on molecule discovery, the process is still defined by a trial-and-error mindset, which does not favor the fact that HTS is immensely costly [Sch+20]. To date, developing new drugs has been a time- and resource-intensive process, primarily driven by a high proportion of failures in clinical trials [Sch+20]. This holds true despite steady advances in technology and methods and a better understanding of disease biology. All in all, biological systems are complex, and identifying suitable candidate molecules requires extensive experimentation and testing, so molecule dis-

covery remains challenging and prone to failure [Bro+20a]. It was reported in 2010 that the cost for pharmaceutical companies to bring a new molecular entity to the market is estimated to be approximately $1.8 billion and the development takes 13.5 years on average [Pau+10]. Although computer simulations have been successfully integrated into this process, allowing virtual screening of molecular libraries, these models usually provide only a mapping between the molecule and its properties [Che+20]. However, modeling the inverse mapping of properties to the molecular structure would enable a targeted search in the molecule space toward promising candidate molecules.

The aforementioned shortcomings have led to the third state of molecule discovery, characterized by increased incorporation of data-driven approaches [Che+20]. In addition to the classic empirical, theoretical, and computational methods, the emerging data-intensive science is discussed as the fourth paradigm of science with a major influence on various fields of research [HTT09; HT20]. Machine learning algorithms are now an essential part of the computational and medicinal chemists' toolbox, driven by the growing amount of digitized molecule data and increasing computational power available [Bro+20a]. In addition to the ever-increasing amount of publicly available data, user-friendly software packages open the field to a broader community of scientists, making it no longer the domain of experts alone. There are multiple freely accessible databases of chemical structures, e.g., PubChem [Kim+21] listing over 110 million unique compounds, ChEMBL [Men+19] with over two million compounds, and ZINC [SI15] a database of 230 million commercially available compounds. This sheer amount of data can pose a problem to classic methods from computational chemistry but opens up possibilities for the application of AI-based methods, especially from the domain of machine learning [Bro+20a].

AI promises improvements to the trial-and-error process of molecule design. For one, AI offers new possibilities to predict a molecule's properties before it has been made in the laboratory [But+18]. Furthermore, these models may not only provide higher predictive accuracy but can also capture the underlying structure of the molecular property space and enable the search for new promising molecules [Bro+20a]. Such an AI modeling the distribution of molecular data can be used to sample new molecule entities, just like sampling a random number from a probability distribution [Pol+20]. These so-called generative models open up a new paradigm of de novo molecular design, i.e., designing molecules from scratch based on the desired properties [Bro+19].

Autonomously generating molecules with AI could obviate the necessity of costly full-deck HTS and drastically reduce the time, cost, and risk associated with developing new drugs and materials by proposing promising and unbiased starting points [Sch+20]. In contrast to lengthy design, test, and experiment cycles that are state-of-the-art in molecule search up to date, these algorithms can traverse the vast search space of potential molecules in a directed and structured manner [Bro+19]. Furthermore, AI research offers a variety of optimization approaches to improve molecules in the direction of desired properties and strategies to deal with constraints like solubility or toxicity [Pol+20].

All in all, de novo molecular design promises to dramatically reduce the number of molecules that need to be evaluated in the design process by targeting promising regions of search space. Conventional methods from computational chemistry struggle with such challenging problems involving massive combinatorial spaces with nonlinear relations. However, AI is regularly applied to multi-objective optimization problems and has the potential to approximate even complex functions by learning from large, unstructured data sets. This renders AI a promising addition to the molecular design process.

In particular in drug design, AI promises some considerable advantages by promising improvements to various parts of the design-make-test-analyse (DMTA) cycle. DMTA is the dominant design process for discovering new drugs. In an iterative trial-and-error manner, a team of experts optimizes molecules toward specific target properties. DMTA commonly involves using theoretical chemistry to propose changes to known structures, synthesizing the candidate molecules, measuring their performance in a series of in vitro and in vivo tests, and analyzing the results. This process can span many years until a suitable candidate molecule has been identified since every hypothesis cycle can take up to 4–8 weeks [Tho+22]. Especially in the early stages of this process, large quantities of hypotheses are tested in a brute-force manner by screening huge molecule libraries with minimal feedback or adjustments. Such a process is not only expensive and inefficient but is also vulnerable to noisy data, personal bias, and poor intuitive choices [Sch18]. Although the DMTA cycle is a central concept in drug design, many of these challenges are transferable to other domains of molecule design.

AI research has proposed solutions targeted at different stages of the DMTA cycle. Retrosynthesis models based on deep neural networks can predict reaction paths for a given molecule and could facilitate the *make* stage, e.g., as demonstrated by Segler, Preuss, and Waller [SPW18]. However, the design part has received the most attention, with various proposed de novo design models [Bro+20a]. These so-called molecule generation models are often trained on large publicly available molecule databases, e.g., ChEMBL, PubChem, and ZINC. Just like random number generators, the models allow the sampling of new molecular structures and can be used to explore the molecule search space. Since these models are trained on libraries of known compounds, the generated molecules have feasible structures and are similar to the training molecules in their property distributions. In drug design, these models are also evaluated for their capability of generating realistic and "drug-like" molecules. In this context, realism is defined as the molecules having a high degree of similarity to known structures. In silico molecule design can be integrated into the design stage of the DMTA and function as a small, virtual DMTA cycle, as pictured in Figure 1.1.

The in silico design cycle seeks to generate promising molecule candidates and optimize them in the direction of the desired target property profiles [Bro+19]. Therefore, molecule generation models can potentially improve the quality of the hypothesis generated in the design phase of drug discovery, which could reduce the time and cost of subsequent steps in the DMTA cycle [Tho+22]. This promise has led to an active research area of molecule generation models, which are also a central research topic of this thesis and of which an overview is provided in Section 3.3.

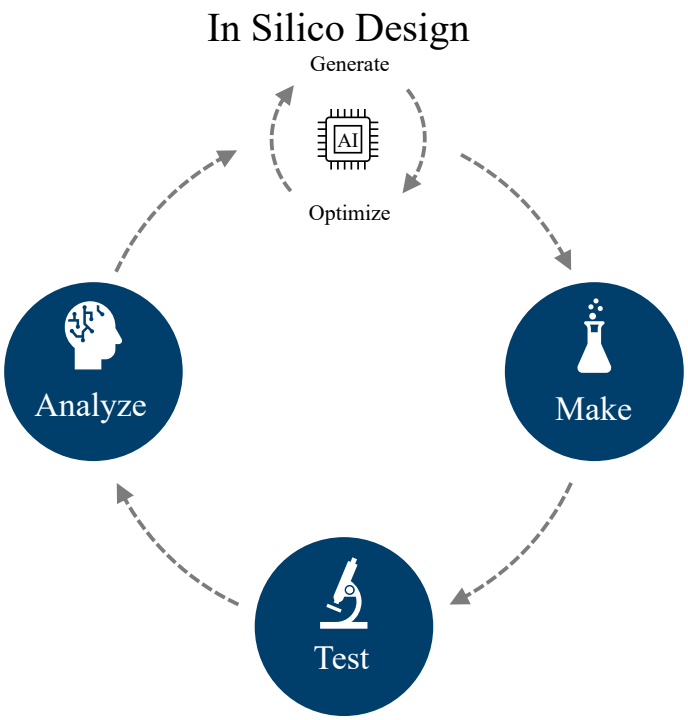


Figure 1.1: Integration of in silico molecule design in the DMTA cycle.

A key challenge in applying AI to chemistry is the question of how to represent the molecule data. Be it predicting molecular properties, generating new compounds, or optimizing existing molecules; computers need a means of storing, processing, and generating molecular data. A proper representation is key to a successful application and should be informative enough to capture the molecular properties relevant to the task at hand but also easy to process by machines. Furthermore, generative models require a representation that is reversible to molecular formulas. Thus, an inadequate representation can be the Achilles' heel of molecule generation [Xue+19].

Figure 1.2 shows the four most common molecule representations for the same exemplary molecule. The following is a brief overview, a more detailed explanation can be found in the respective chapters of this thesis. Figure 1.2a pictures a representation that has been in use for a long time: molecular fingerprints. These often fixed-size bit vectors encode the structural features of a molecule. Various types of fingerprints are available, each with a specific set of rules determining how the bit string is constructed. Further information on fingerprints is provided in Chapter 7. For a long time, fingerprints have been a dominant representation in computational chemistry, as they allow a straightforward way of comparing the similarity of molecules, which makes them especially applicable for virtual screening [Pol+20].

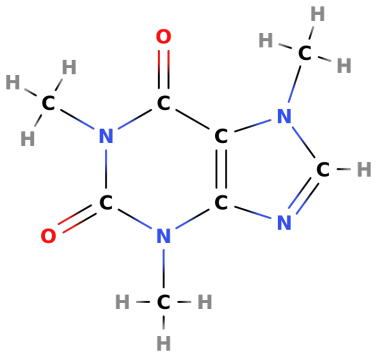
One of the most common types of molecular representations today is string-based representations, encoding the molecular graph as a sequence of characters. A prominent example of string-based representations is the simplified molecular-input line-entry system (SMILES) [Wei88], as depicted in Figure 1.2b. SMILES specifies a grammar and syntax rules to encode a molecule spanning tree, and a SMILES string contains characters specifying a molecule's

. . . 0000001000000001000000. . .

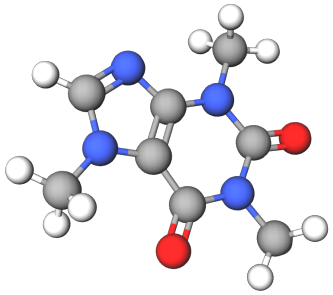
(a) Molecular fingerprints

CN1C=NC2=C1C(=O)N(C. . .

(b) String-based



(c) Graph-based



(d) Spatial

Figure 1.2: Four common types of molecular representation, illustrated using an exemplary organic molecule.

atoms and their bonds. It also includes special characters to signalize more complex concepts like branches, ring structures, or stereochemistry. String-based representations are simple to process in terms of storing, reading, and writing. Furthermore, they allow a straightforward adaption of language modeling tools, e.g., recurrent neural networks, to be applied to molecule generation [Pol+20].

Operating directly on a molecule's graph representation has become more prevalent in recent years due to the increased knowledge of graph-modeling techniques [GMH22]. A graph is an abstract data structure of nodes, specifying an entity, and edges between the nodes, representing relations between the entities. In terms of molecular graphs, each node in a graph specifies an atom of a specific type, while edges represent bonds between these nodes (see Figure 1.2c). This way of encoding a molecule's structural information in a more mathematical way can benefit neural networks and eliminates the need for the model to learn a specific syntax, as is the case for string-based representations.

There are also examples of models operating on three-dimensional representations of molecules, as pictured in Figure 1.2d. This representation provides access to the spatial information of the molecule and allows distinguishing between spatial isomers of the same structural formula (the concept of isomers will be featured in Section 3.1). However, creating molecules directly in three-dimensional atom space can be a difficult task, as it requires the prediction of precise atomic distances and angles to ensure that the molecule is valid.

All the presented representations offer their advantages but also challenges when applied to the generation of molecules. This work presents new methods for generating and optimizing molecules in all four aforementioned representations. We will investigate state-of-the-art AI-based molecule generation models and discuss their possibilities and limitations. Furthermore, we will introduce new AI techniques for generating drug-like molecules that will open

up new opportunities and applications. In this regard, we will focus on the recently introduced transformer architecture for sequence processing. Transformers are a unique neural network architecture initially designed for language processing tasks. Their unique attention mechanism allows them to effectively process sequential information, as it allows attending to arbitrary points in a target sequence. This benefit makes them an alternative to the methods used so far, which are usually based on recurrence. In addition, their high degree of parallelism allows training models with a considerable number of parameters, making them capable of capturing even complex relationships. This work discusses the modeling and strategies required to apply this powerful architecture to the field of molecule generation and highlights the advantages that such an application offers. The following sections provide a brief overview of this thesis' structure and the publications on which this work is based.

1.1 Overview of the Thesis

An overview of this thesis' structure is pictured in Figure 1.3. The thesis is divided into three main parts. Part I provides an overview of the theoretical foundations from AI, chemistry, and molecule design on which this thesis is based. Part II presents four contributions to how AI can be utilized to generate drug-like molecules in the four aforementioned representations. Part III summarizes the results, presents a conclusion, and provides an outlook for possible future work.

The foundations (Part I) begin with an introduction to the core concepts of AI on which the research in this thesis is built in Chapter 2. This includes the concept of optimization and the fundamentals of deep learning, with a particular emphasis on deep generative models. Building on this, Chapter 3 pictures how these concepts have been applied to molecule design and optimization thus far. The chapter starts with an introduction to the basic concepts of organic chemistry. After that, it provides an overview of common molecular representations and typical applications of AI in the context of molecule design. The chapter concludes with an overview of the implications of AI for molecular design and the ethical implications that arise from such a development.

Part II consists of four research contributions, each focusing on a different molecule representation. In the first study presented in Chapter 4, a genetic algorithm (GA) is applied to the generation of molecules in the string-based SELFIES representation. The molecules are designed to function as SARS-CoV-2 protease inhibitor candidates and are evolved in a multi-objective approach. Section 4.1 explains the concept of evolution-inspired algorithms such as GAs, whilst Section 4.2 gives a brief overview of the problem domain: protease inhibition. The evolutionary multi-objective approach for inhibitor design is introduced in Section 4.4. The chapter concludes with an analysis of the generated molecules and their potential performance as protease inhibitors.

Generative models based on neural networks can facilitate the generation of drug-like molecules. The approach presented in Chapter 5 demonstrates how a neural network based on transformers can be used to generate molecules in a spatial representation. The chapter briefly

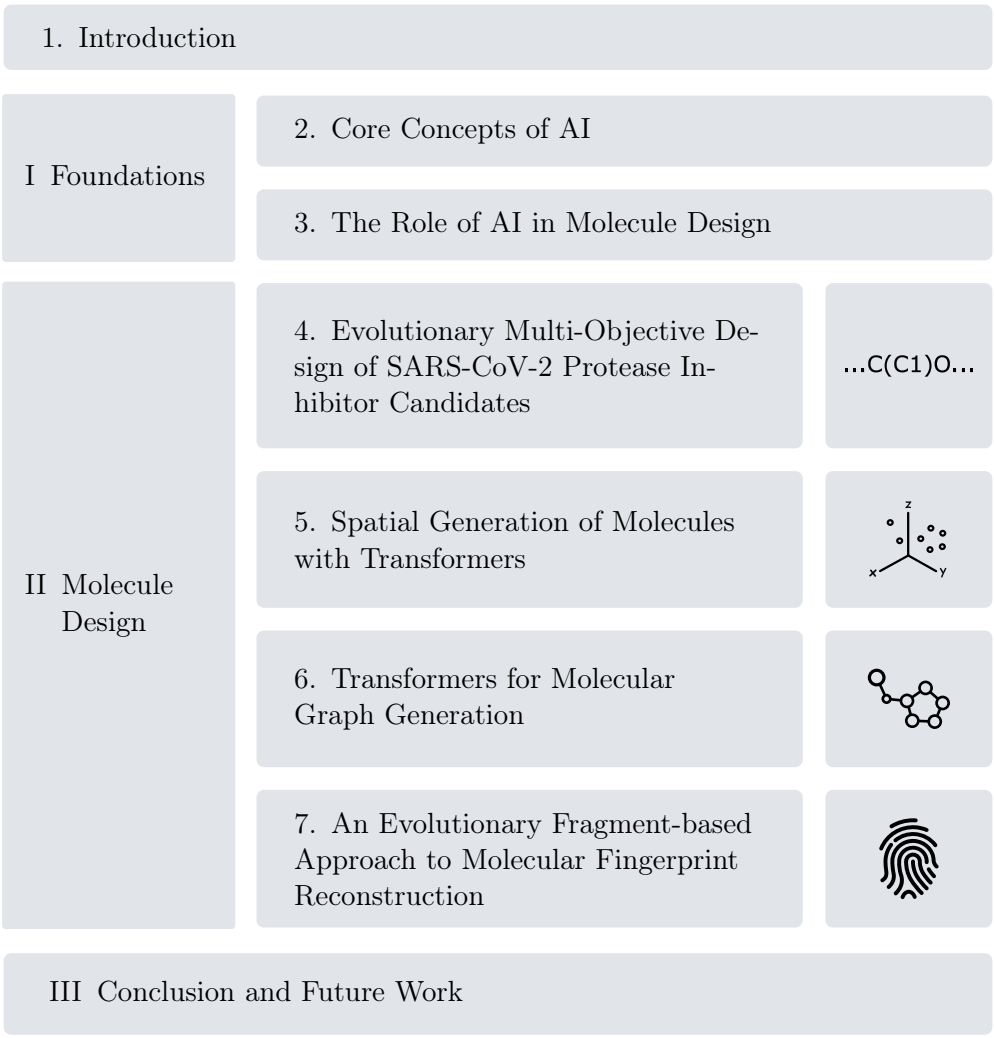


Figure 1.3: Structure of this thesis.

introduces the transformer architecture in Section 5.1. Building up on this, the study explores the difficulties arising when sampling molecules directly in three-dimensional space and the adjustments that need to be made to the transformer architecture to make this possible. Generating spatial molecules with transformers requires a customized way of representing the molecular information but offers advantages for training and sampling, which are all discussed in Section 5.3. A further advantage of the proposed approach is the capability of completing unfinished, unconnected atoms to valid molecular structures. This property is investigated in Section 5.4, as is the general performance of the model in generating new spatial molecules.

Building upon these results, Chapter 6 improves the processing and sampling strategy introduced in the previous approach and transfers it to the domain of molecular graph generation. This is achieved by incorporating state-of-the-art strategies from graph synthesis into the model, as described in Section 6.2. The model's capabilities of generating valid and novel molecules are experimentally investigated in Section 6.3, in which the model is also compared to other prominent molecule generation models.

Chapter 7 addresses the topic of molecular fingerprints as a representation for generation models. Since these bit string representations are usually not decodable, they are rarely used for molecule generation. However, the approach presented in this chapter demonstrates how a GA can be utilized for molecular fingerprint reconstruction. Section 7.1 describes how the algorithm combines molecule fragments into valid molecules and the employed genetic operations. The algorithm's reconstruction capabilities are investigated in Section 7.2. Furthermore, a transformer-based model for fingerprint generation is introduced, which complements the GA to a molecular generation model based on fingerprints.

All in all, these four contributions demonstrate new techniques for molecule generation models for the four most prominent molecule representations in computational chemistry. Chapter 8 concludes this thesis by discussing the introduced enhancements to molecule generation in different representations. Furthermore, the two main AI instruments featured in this thesis, namely GAs and transformers, are evaluated in their utility as molecular generation models. Finally, a summary of possible future work is provided.

1.2 Contributions

This thesis features four main contributions in Part II published as peer-reviewed conference articles. A short overview of the publications is presented in the following.

- The results in Chapter 4 are based on a collaborated work and demonstrate how a genetic algorithm can be utilized for the generation of protease inhibitor candidates. The approach represents molecules in the string-based SELFIES representation and is a multi-objective optimization, i.e., it considers five molecule metrics. The results are published in:
  - T. Cofala, L. Elend, P. Mirbach, J. Prellberg, T. Teusch, and O. Kramer. "Evolutionary Multi-objective Design of SARS-CoV-2 Protease Inhibitor Candidates". In: *Parallel Problem Solving from Nature - PPSN XVI - 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part II*. ed. by T. Bäck, M. Preuss, A. H. Deutz, H. Wang, C. Doerr, M. T. M. Emmerich, and H. Trautmann. Vol. 12270. Lecture Notes in Computer Science. Springer, 2020, pp. 357–371. DOI: 10.1007/978-3-030-58115-2_25
- Chapter 5 presents an approach for generating molecules in a three-dimensional representation. The approach is based on transformers and is able to either generate new molecules from scratch or even link predefined molecule fragments. The results are published in:
  - T. Cofala, T. Teusch, and O. Kramer. "Spatial Generation of Molecules with Transformers". In: *International Joint Conference on Neural Networks*. IEEE, July 2021, pp. 1–7. ISBN: 978-1-6654-3900-8. DOI: 10.1109/IJCNN52387.2021.9533439.

- The results in Chapter 6 extend the previous approach for the generation of molecular graphs. A more efficient sampling procedure based on mixed categorical distributions allows the generation of longer molecules. The results are published in:
  - T. Cofala and O. Kramer. "Transformers for Molecular Graph Generation". In: *ESANN 2021 proceedings*. Louvain-la-Neuve (Belgium): Ciaco - i6doc.com, 2021, pp. 123–128. ISBN: 978287587082-7. DOI: 10.14428/esann/2021.ES2021-112
- In Chapter 7 a novel strategy for the reconstruction of molecular fingerprints is introduced. The algorithm is based on GAs and composes molecules from a set of fragments. Combining the approach with a transformer-based generation model for molecular fingerprints offers new possibilities for molecular design. The results are published in:
  - T. Cofala and O. Kramer. "An evolutionary fragment-based approach to molecular fingerprint reconstruction". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, July 2022, pp. 1156–1163. ISBN: 9781450392372. DOI: 10.1145/3512290.3528824

Part I

Foundations

2 Core Concepts of AI

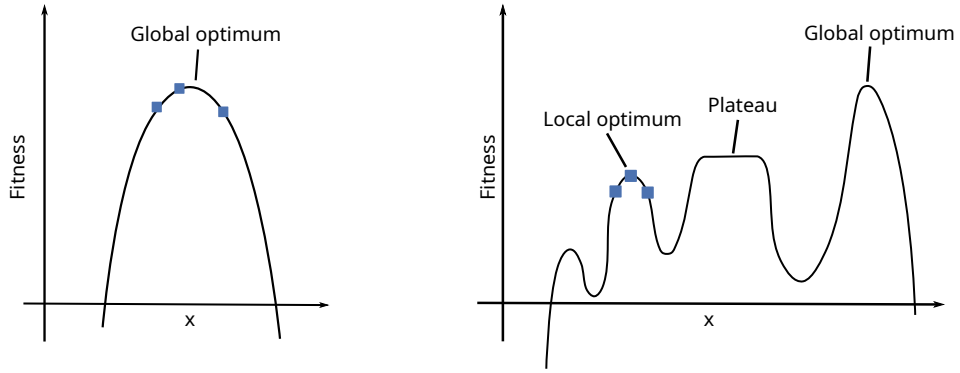
AI is a research domain with many facets. In fact, so many different families of algorithms are subsumed under this term that it is difficult to find a unified definition. Russell and Norvig [RN16] call AI a field dedicated to build and study intelligent entities. Since it is generally difficult to give a clear definition of what makes an entity intelligent, they focus on the concept of rational agents. An agent is an entity that lives in an environment, perceives that environment, and acts according to a specific goal, adapting its behavior in response to change. A rational agent seeks to interact with the environment to maximize the expected outcome with respect to that goal. AI deals with the construction of such rational agents.

Just as there are many facets to human intelligence, the development of intelligent agents has many sub-fields. Research focuses on various domains that impact the agent's ability to achieve its goal, such as problem-solving, reasoning, planning, learning, perceiving, acting, and more. This section provides an overview of AI concepts that form the basis for the research in this study. Section 2.1 introduces the concept of problem-solving, with a focus on optimization problems. In Section 2.2, we address how an agent can learn from data to improve its performance in the given task. Since this work concerns the generation of new molecules, Section 2.3 pays special attention to deep generative learning.

2.1 Problem-solving and Optimization

Optimization problems are a common problem domain, with examples from various areas of our everyday life. A central element of these problems is the objective function (also called fitness function), a quality measure of the generated solutions. Typical objectives are, e.g., minimizing an error, energy consumption, weight, or cost and maximizing profit, outcome, or success [Kra17]. Optimization algorithms aim to generate valid solutions for the given problem with optimal scores concerning the objective function.

For most problems, the space of potential solutions is so vast that a complete exploration is infeasible. Therefore, an optimization algorithm has to search the solution space effectively to find an optimal or near-optimal solution. The shape of the objective landscape has a significant impact on the complexity of the optimization problem. Two examples of objective landscapes are pictured in Figure 2.1. The x-axis represents the variable to be optimized, while the y-axis shows the corresponding performance with respect to the objective function. The unimodality of the objective landscape in Figure 2.1a makes it easy to find the best solution, the global optimum. Usually, objective landscapes have a more complex shape, with many plateaus and local optima, as shown in Figure 2.1b. A local optimum has a



(a) Unimodal (b) Multimodal

Figure 2.1: Objective landscapes of optimization problems.

higher objective score than its surrounding but less than the global optimum. An efficient optimization algorithm has to be able to deal with multimodal objective landscapes to avoid getting trapped in such local optima. Furthermore, most optimization problems require the optimization of multiple variables, rendering the objective landscape even more complex.

The topic of optimization will be explored in more depth in later chapters of this thesis, where it will be applied to the design of molecules. Chapter 4 will introduce GAs, a popular optimization algorithm, and demonstrate how it can be utilized to optimize molecules as drug candidates. In Chapter 7, a GA is applied to the reconstruction of molecular fingerprints.

2.2 Learning

Learning is the second central concept of AI relevant for this thesis. Learning helps an agent improve its performance for a given task by observing data from the real world [RN16]. This approach is especially useful when the problem's underlying structure and the required solution strategy are not known to the designer [RN16]. Three main components define learning algorithms: A *task* in which the algorithm tries to improve with respect to a certain *performance measure* by learning from *experiences* [GBC16]. Machine learning can be applied to a variety of different tasks. Common problem domains are, e.g., classification, regression, anomaly detection, machine translation, and many more [GBC16]. The performance measure usually depends on the problem domain, with common ones being accuracy or error rate. Machine learning aims to build a model that makes predictions for the given tasks, maximizing these performance measures. A key challenge in this regard is that the model should be able to generalize from observed experience and be applicable to previously unknown inputs. A model, therefore, approximates a function that maps a given set of inputs to hopefully reasonable predictions or decisions.

Depending on the structure of the observed experiences, i.e., the data, machine learning can be broadly divided into three categories: Supervised, unsupervised, and reinforcement learning. In supervised learning, each data point is associated with a concrete label. The

training objective is to learn a mapping from a given input to a probability distribution over the possible labels. A model trained in such a fashion can, e.g., be utilized for classification tasks. Unsupervised learning, on the other hand, does not include any labels and focuses on the structure of the data itself. Its objective is to approximate the probability distribution underlying the data, which can then be used for synthesis or denoising [GBC16]. A further common application domain of unsupervised learning is clustering by identifying groups of similar data points. The last type of learning is reinforcement learning, which is driven by feedback from an environment. Based on the observed experiences, the algorithms interact with their environment and are rewarded based on the performance. Typical applications are learning robot control or strategies for games.

The capacity of the model has a significant influence on its possible achievable performance. Capacity describes the variety of functions a model is able to fit [GBC16]. A model's capacity increases with its number of adjustable parameters and the variety of functions it can combine to construct its outputs. Choosing the right capacity for a model can be challenging, but the development of a model's performance during training usually provides a good indicator. Performance is typically measured against a test set that includes data not presented to the model during training. This test error shows how well the model can generalize to unseen data. Especially the relationship between training error and test error indicates if the chosen model and its capacity are suitable for the specific problem. If the capacity is too low and the given task too complicated, underfitting can occur, indicated by a high training error. An underfit model is simply not powerful enough to capture the structure of the data and cannot be used to make meaningful predictions. Conversely, a model with a too-high capacity can result in overfitting, indicated by a low training error but a high test error. Especially if there is not enough training data available, the high complexity of the model may cause it to capture the training data too accurately, resulting in poor generalization. All in all, a model performs at its best if its capacity matches the complexity of the given problem and a sufficient amount of data is provided [GBC16].

Many classical machine learning models struggle with the high dimensionality of tasks like speech recognition or image processing. However, with the continuous improvement of deep neural networks, machine learning became applicable to many complex problem domains, and deep learning was established. A neural network is composed of units that loosely resemble a neuron in neuroscience. Neurons receive inputs from other neurons and combine them into a scalar output. Every neuron, therefore, implements a linear vector-to-scalar function. The neuron has a set of weights multiplied by the inputs and summed to construct the output. This computation can be facilitated by grouping the neurons in layers and processing them in parallel. Combined with an input-independent bias, a layer represents a vector-to-vector mapping of the form:

$$f(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b} \tag{2.1}$$

The weight matrix $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is multiplied by the input vector $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$ summed to the bias $\mathbf{b} \in \mathbb{R}^{d_{out}}$. This most straightforward type of layer is called a feedforward, linear or dense layer and forms the backbone of most deep neural networks. A layer is usually followed by

a nonlinear activation function employed element-wise to allow the approximation of more complex function types. A common example of such an activation function is the rectified linear unit (ReLU):

$$\mathrm{ReLU}(\mathbf{x}) = \max(0, \mathbf{x}) \tag{2.2}$$

A deep neural network with multiple subsequent layers can offer an immense number of parameters θ, e.g., the weights and biases. By adjusting these parameters, the network can be trained to approximate an arbitrary target function $f^*(\mathbf{x})$ [GBC16]. During training, the network observes examples of inputs $\mathbf{x}$ and outputs $\mathbf{y}$ produced by the target function $f^*(\mathbf{x})$. The network defines a mapping $y = f(\mathbf{x}; \theta)$ that is iteratively improved to match the real function by updating the parameters θ [GBC16].

Of course, this is a relatively high-level overview of the broad field of deep learning. An in-depth explanation of the underlying principles, the mechanism of training through stochastic gradient-descent, and various techniques to improve training stability and predictive power can be found, e.g., in the work of Goodfellow, Bengio, and Courville [GBC16].

2.3 Deep Generative Learning

The previous section introduced machine learning, a family of algorithms with a wide range of possible applications. One of these applications is generating new data points based on conclusions drawn from the training data. This section provides an overview of generative learning, as it is a crucial concept on how AI can be utilized to generate novel molecules.

Foster [Fos19] describes generative learning as follows: A generation model aims to generate realistic instances of a type of entity. For example, its objective could be to generate realistic images of a specific object. For this purpose, the model is trained on a dataset of real instances of the target class. The previous section already introduced the concept of training a machine learning model for function approximation. In generative learning, the model is trained to approximate the unknown probability distribution that generated the training data p_{data}. In contrast to classification, where the goal is to predict a label y for the data point $\mathbf{x}$, $p(y|\mathbf{x})$, generative learning estimates the probability of observing the $\mathbf{x}$ at all: $p(\mathbf{x})$. The probability can also be conditioned on a label if one is present $p(\mathbf{x}|y)$. Accordingly, generative models are necessarily probabilistic models. A model trained in such a fashion can then sample data instances that are realistic in the sense that they are also likely in p_{data}. As an additional objective, many generative models aim to generate new instances, i.e., they are suitably different from the training data. In this manner, the generative model can thus be applied to the search for new but realistic instances of the respective target class.

Every observation instance consists of features, for example, pixels in an image. All observations exist in a sample space spanned by all possible values these features can take. The function p_{data} and its by the model approximated function p_{model} are considered probability density functions, i.e., they map a point from the sample space to a value between zero and one, and the sum of all probabilities from all points in the sample space is one. There are many possible p_{model} that, based on the observations in the training data, estimate the true

p_{data}. One way of constructing a suitable approximation is using a parametric model $\hat{p}_\theta(\mathbf{x})$ that defines a probability density function based on a finite set of adjustable parameters θ.

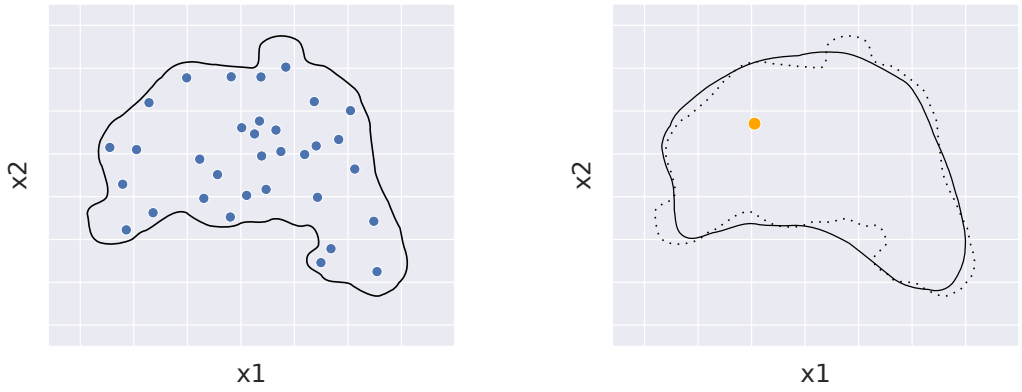


Figure 2.2: On the left: Sample data points and the true data-generating distribution $p(\mathbf{x})$. On the right: The estimated distribution $\hat{p}_\theta(\mathbf{x})$ and one exemplary generated point.

Figure 2.2 visualizes an exemplary two-dimensional space with a set of observations represented as blue dots. The observations are generated by the unknown distribution $p(\mathbf{x})$ marked in blue. The distribution on the right is the approximation $\hat{p}_\theta(\mathbf{x})$ that is fit to match the $p(\mathbf{x})$ by adjusting the parameters θ. With the help of $\hat{p}_\theta(\mathbf{x})$, new observations (e.g., the orange dot) can be sampled that should also be realistic with respect to $p(\mathbf{x})$.

As discussed in the previous section, this can, e.g., be achieved with a deep neural network resulting in the domain of deep generative learning. The training objective is usually defined as the likelihood $\mathcal{L}(\theta|\mathbf{x})$ of θ for a given observation $\mathbf{x}$. For a set of independent observations $\mathbf{X}$, the likelihood defined as the product of the individual probabilities $\mathcal{L}(\theta|\mathbf{X}) = \prod_{\mathbf{x}\in\mathbf{X}} p_\theta(\mathbf{x})$. Usually, the log-likelihood $l(\theta|\mathbf{X}) = \sum_{\mathbf{x}\in\mathbf{X}} \log p_\theta(\mathbf{x})$ is used to avoid the calculation of the product.

Although research primarily focuses on classification models, partly due to their high degree of comparability, there are various generative models for a variety of data types. Famous examples from the domain of language processing are the text-generating model generative pre-training (GPT)-3 [Bro+20b] and WuDoa, capable of writing articles, poems, source code, and many more. Jukebox is a generative model for music introduced by OpenAI that can generate various styles of music and even parts of rudimentary singing [Dha+20]. In the domain of images, DALL·E, currently in its second version, is able to generate photorealistic images from text queries [Ram+22].

3 The Role of AI in Molecule Design

With its tremendous capabilities to solve even complex optimization tasks and its ability to generate innovative solutions, AI has found its way into many areas of industry and research. This is also true for the field of chemistry, where AI shows promise in reducing the need for complex simulations and facilitates the design of novel molecules. This chapter provides a brief introduction to organic chemistry in Section 3.1 and typical applications of AI in this field in Section 3.3. For this purpose, Section 3.2 pictures an overview of machine-friendly molecule representations, which are a prerequisite to process molecules with AI efficiently. The chapter concludes with a summary concerning the applications of AI in molecular design presented in Section 3.4 and the ethical implications that arise from such integration in Section 3.5.

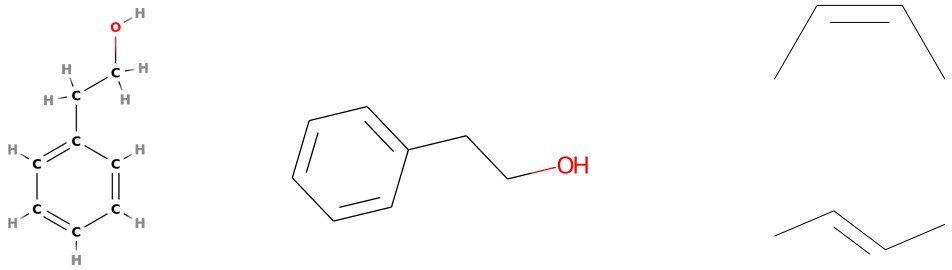
3.1 Organic Chemistry

The molecules generated and optimized in this thesis belong to the field of organic chemistry. This section provides a brief overview of the fundamental concepts of organic chemistry and the characteristics of such molecules. However, this overview only scratches the surface of this vast field of science. For a deeper insight, the reader is kindly referred to the literature from which the following information is obtained ([All+80; Liu21]).

Organic chemistry deals with molecules based on the element carbon, from which all living things are built. Initially, the term only referred to molecules derived from living organisms and was introduced in 1807 by Jöns Jakob Berzelius. However, the definition shifted to considering all those molecules based on the strong covalent C–C and C–H bonds. Based on this definition, not all molecules containing carbon are considered organic, but organic compounds still form all living organisms.

Molecules are structures of atoms that are held together by attractive forces, so-called bonds. Chemistry considers two major types of bonds between atoms, namely ionic and covalent bonds. Ionic bonds are based on electrostatic attraction and occur between ions of opposite charges. Covalent bonds, on the other hand, are the result of two bonding atoms sharing electron pairs. Organic chemistry mainly considers molecules composed of atoms with covalent bonds. Thereby, the concept of valence plays an essential role in this type of bond. An atom's valence electrons are those electrons that are on its outermost shell. Since these electrons are the furthest from the atom's nucleus, they are least attracted by its forces. Valence electrons are the most reactive electrons of an atom and have the most significant influence on bonding. The amount of valence electrons of a molecule defines the number of

bonds it can form with other atoms. By sharing electron pairs, the two bonding atoms try to achieve a filled outer shell, i.e., an octet.



(a) A molecular graph and the corresponding skeletal formula representation.

(b) The same molecule in a cis and trans conformation.

Figure 3.1: Exemplary organic molecules.

The carbon atoms of an organic molecule form strong covalent bonds in the form of chains and rings. These structures build the carbon backbone of every organic molecule. Carbon has a valence of four, which means it strives to form four bonds with other atoms. In their simplest form, organic molecules only consist of these carbon structures filled with hydrogen based on the remaining valence. Even when only considering these simple hydrocarbons, the number of potential organic molecules is theoretically infinite. However, carbon is also capable of forming strong bonds with other elements like oxygen, nitrogen, or sulfur, resulting in a diverse and complex space of potential molecules. Since hydrocarbons are generally relatively inert, the incorporation of other atoms often has a significant effect on the molecule's properties. Especially the functional groups—common structures of specific atoms—determine the chemical reactions an organic molecule can undergo. Figure 3.1 depicts some exemplary organic molecules. Figure 3.1a shows a molecular graph and the corresponding short-line structure representation (also known as skeletal formula). In this representation, each line expresses a bond between two atoms. Carbon atoms are usually omitted, and the carbon chains are instead shown in a zigzag pattern. Each end of a bond and each intersection represents a carbon atom unless a different atom type is explicitly shown. As long as hydrogen atoms are bound to carbon atoms, they are usually not pictured. 2-Phenylethanol, the molecule pictured in Figure 3.1a, features a hydrocarbon skeleton with a carbon ring and a small carbon chain. At the end of the carbon chain is a hydroxyl group, which leads to the molecule being classified as an alcohol.

Figure 3.1b shows the same molecule but highlights an additional important concept in organic chemistry: isomers. Isomers share a chemical formula but differ in structure. There are two types of isomers: Stereoisomers, which show the same bonding but differ in the spatial arrangement of atoms, and constitutional isomers, which differ in the way atoms are connected. Although stereoisomers share the same bonds between atoms, the spatial arrangement can significantly impact the molecule's properties. Stereoisomers are further distinguished be-

tween geometric isomers, isomers containing a chirality center, and conformational isomers. Geometric isomers are divided into two groups depending on whether parts of the molecule are either on the same side, “cis”, or on opposite sides, “trans”, of a reference structure, such as a carbon ring or a double bond. An example is pictured in Figure 3.1b. Two molecules are chiral if they are mirrored but can not be superimposed, i.e., transferred to each other by rotation and translation. These isomers usually occur in molecules containing a stereocenter, such as a carbon with four attached groups. Finally, conformational isomers are molecules with the same chemical formula but different spatial arrangements caused by single-bond rotations. These different configurations are called conformations, and, in contrast to previous stereoisomers, molecules can change their conformation if a high enough energy is applied. However, different conformations usually exhibit a varying degree of stability and thus occur more frequently.

3.2 Molecular Representations

There is a variety of possible representations for molecular structures, for example, the already in the previous section mentioned short-line structure. In general, every representation has its characteristics in the form of simplicity, expressiveness, and ambiguity. This section provides a high-level overview of the main molecule representations in computer science. A more in-depth overview of approaches based on the respective representations is provided in the corresponding research chapters 4–7.

3.2.1 Graphs

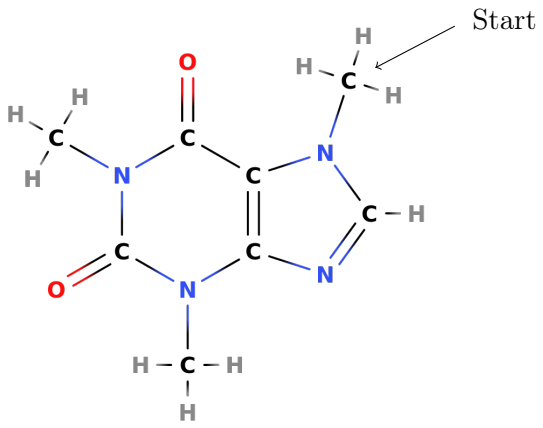
Graphs are a common data structure in computer science and can be applied in various domains. A graph consists of nodes and edges. Nodes represent entities and can be connected to other nodes through edges, displaying a relationship between the two nodes. Nodes and edges can feature additional information to describe the objects and their relations further.

In chemistry, graphs can be used as a means of representing a molecule, as shown in Figure 3.2. A molecular graph contains nodes corresponding to its atoms and edges representing the bonds. The nodes contain information about the atom type but can also feature further information, like hybridization or charge [GMH22]. Edges usually contain information about the respective bond type [GMH22].

Representing molecules as graphs is an effective way to encode the topology of a molecule in a mathematical format. This allows the adaptation of specialized types of neural networks, such as graph convolution networks or message passing neural networks, for processing molecules [Pol+20]. However, graphs are a more abstract representation that can complicate the interpretation of results [GMH22]. Moreover, graphs can usually not express differences between stereoisomers of the same molecule, as spatial information is not captured in graphs.

3.2.2 Strings

The introduction of computers into chemistry caused the development and widespread application of string-based molecular representation. Computers can process linear strings with ease [Wei88] and string representations offer a condensed way of storing vast amounts of molecular data.



Possible SMILES:
CN1C=NC2=C1C(=O)N(C)C(=O)N2C

Canonical SMILES:
CN1C=NC2=C1C(=O)N(C(=O)N2C)C

Figure 3.2: A molecular graph with one of the possible SMILES strings and the canonical SMILES representation.

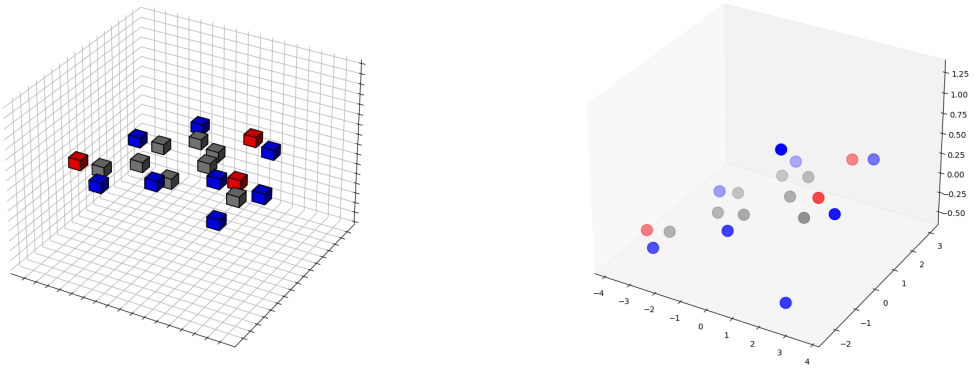
One of the most widely used string representations for molecules are SMILES strings [Wei88]. SMILES defines a vocabulary of symbols, a grammar, and syntactic rules which allow encoding molecular structures into a short sequence of characters. The molecule's graph is the basis for constructing a SMILES string. For simplicity and readability, SMILES has no intention of capturing three-dimensional information about the molecule. Figure 3.2 pictures an exemplary molecular graph and its SMILES representation. The syntactic rules for constructing a SMILES string are briefly described in the following: Atoms are encoded by their atomic symbols. Hydrogens are usually not shown explicitly, and each atom is assumed to hold as many hydrogen atoms as correspond to its valency. Single bonds can be described by the symbol –, but are also usually omitted. Double and triple bonds are explicitly stated and represented by = and #, respectively. Branches are specified by including the branch in parentheses, which can also be nested. Finally, to encode cyclic structures, a ring is broken at a single bond, and the ring opening is marked with a digit. The ring is then traversed in an arbitrary order, and the closure is marked with the same digit. Aromatic rings—common, stable structures in organic compounds—can either be expressed as alternating single and double bonds or by writing the atoms in the aromatic structure as lowercase letters. Furthermore, SMILES contains special characters to capture many more niche cases, but the aforementioned rules are sufficient to express most organic structures. Although there are multiple different possible SMILES strings, SMILES features rules for a canonical ordering to remove ambiguity. To a certain degree, SMILES also offers ways of distinguishing between stereoisomers: The direction of single bonds in relation to a double bond can be expressed with \ and /. Chiral molecules can be differentiated with the @ symbol.

There are further examples for string-based representations, e.g., the international chemical identifier (InChI). The InChI is a unique molecular identifier containing multiple layers of information about the encoded molecule. However, SMILES strings offer advantages over other string-based representations, which facilitated its widespread application [GS22]: SMILES offers a small but flexible set of syntactic rules. This can make it easier for generative models to capture these rules and apply them for the generation of new structures. Additionally, SMILES strings are easily readable and interpretable by humans. These advantages have led to the creation of enormous molecule databases encoded as SMILES strings, which are the foundation of many state-of-the-art generative molecule models. The sequential nature of SMILES strings allows a straightforward application of techniques from natural language processing, which makes SMILES especially suitable for molecule generation.

Despite the ease of use, string-based representations also have drawbacks. For example, they do not contain complete information about the locality of atoms. Furthermore, when generating SMILES strings, it is possible to generate strings that are syntactically not correct. Even if a string is correct, it can encode a molecule with incorrect valences. Self-referencing embedded strings (SELFIES), a further string-based representation, tries to overcome this disadvantage and will be featured in Chapter 4.

3.2.3 Spatial

Representing molecules as graphs or strings bears the disadvantage of ignoring spatial information about the molecules. However, how a molecule interacts with other substances is often of interest, and these interactions take place in a three-dimensional space [GMH22]. Furthermore, in contrast to the other representations mentioned, a spatial representation enables complete distinction between stereoisomers of the same molecule. Although stereoisomers yield the same graph representation, they can have significantly different properties.



(a) Grid-based (b) Point cloud

Figure 3.3: The molecule vanillin in two different three-dimensional representations.

Overall, fewer examples of approaches for property prediction or molecule generation operate on a three-dimensional representation. A significant challenge is representing the molecule in a way that machines can efficiently process. One possible way is rendering the molecule's atoms in a grid of voxels. A voxel is the three-dimensional counterpart of the two-dimensional pixel. In such a voxelized coordinate system, every voxel is either empty or contains information about an atom. An example of the molecule vanillin is given in Figure 3.3a. This representation is quite flexible and not only allows including further information about an atom in additional channels of the grid space but can also contain parts of a target structure, e.g., a protein's binding site [Pol+20]. Finding a molecule that binds to a protein is a common task in drug design and will be featured in Chapter 4. However, a disadvantage of grid-based approaches is the sparsity of the encoded space, i.e., usually, most voxels are empty [Pol+20].

Another way to encode a molecule in 3D is to consider it a point cloud of atoms. Each point is assigned an atom type which, in combination with its distance to all other atoms, is enough information to fully describe a molecule, as shown in Figure 3.3b [GGS19]. This representation, therefore, has no necessity for abstract concepts like bonds or rings [GGS19]. Working with point clouds involves additional complexity compared to grid-based approaches, as the computational model must be able to handle inputs and outputs of varying size and shape.

3.2.4 Fingerprints

Fingerprints are a somewhat abstract molecular representation with a long history in computational chemistry [Pol+20]. There are various types of fingerprints available, but usually, they share a common concept: The structural features of a molecule are encoded in a fixed-length sequence, often a bit string or a count vector. Fingerprints are commonly applied in virtual screening, as they allow an easy and fast way of comparing molecule similarities [RL13]. The various types of fingerprints differ in the set of rules they use to encode the molecule's features and, according to Riniker and Landrum [RL13], can be divided into four categories:

Dictionary-based. A common representative of dictionary-based fingerprints is the public Molecular ACCess System (MACCS) structural keys. These offer a predefined set of 166 substructures. A to-be-encoded molecule is searched for these substructures, and, if present, the corresponding bit in the fingerprint bit vector is set. This procedure produces a fingerprint that is easy to interpret, in contrast to the following types of fingerprints. However, it can only encode the occurrence of predefined features and usually show a low performance level in virtual screening [RL13].

Topological or path-based fingerprints consider all subgraphs of a molecule up to a predefined maximum length. The information about the subgraph, e.g., atom types, aromaticity state, and bond types, is hashed, and the hash is used to set a bit in the bit

vector. A typical representative of path-based fingerprints is the daylight fingerprint introduced by the commonly used cheminformatics software toolkit RDKit[1].

Circular fingerprints consider the neighborhood of every atom of a target molecule. For every atom, the environment up to a particular bond radius is encoded with a hash function and folded into a bit vector of the defined size. Circular fingerprints, like the extended-connectivity fingerprint (ECFP), are a more recent development but have quickly become a popular type of fingerprint.

Pharmacophore fingerprints focus on the occurrence of pharmacophore features in molecules. These features usually play an important role in the binding process. Pharmacophore fingerprints consider the bond distances between pharmacophores and incorporate this information into the fingerprint.

In summary, molecular fingerprints are an easily processable and comparable molecule representation. Although fingerprints are broadly applied for virtual screening and sharing molecule information without revealing the exact chemical formula, there are few examples of fingerprints used for molecule generation. Dictionary-based fingerprints are limited in the number of features they can encode, and fingerprints based on hashing are non-decodable.

3.3 Applications of AI in Molecule Design

With the advancements in representing molecular data in a machine-friendly manner, as discussed in Section 3.2, AI algorithms became applicable to a variety of molecule design problems. This section provides an overview of these application domains.

Quantitative Structure-Activity Relationship. An important aspect of identifying suitable candidate molecules for drug development is the reliable assessment of a molecule's properties. Quantitative structure–activity relationship (QSAR) models are computational tools that capture the relationship between a molecule's descriptor and its biological activities. Xu [Xu22] provides an overview on how QSAR models are applied in the drug design process. For one, QSAR models can be utilized to prioritize candidate molecules for the given task and evaluate the quality of newly generated compounds. Furthermore, they can provide insight into how structural changes in a molecule affect its biological activities. Since the development of QSAR models amounts to the solution of regression and classification tasks, AI methods have been frequently used in this area. The models are usually trained using experimentally obtained data on molecular properties. Especially deep neural networks have proven to be powerful tools for these tasks, as, e.g., demonstrated by Ma, Sheridan, Liaw, Dahl, and Svetnik [Ma+15]. Using deep neural networks, they won a Kaggle competition in 2012 to find state-of-the-art machine learning methods for QSAR, beating the previously dominant random forest approaches.

[1] https://www.rdkit.org

QSAR models based on deep learning can significantly reduce the time needed to test a large number of compounds. Commonly, these models operate on either string-based representations, mostly SMILES, or fingerprints, like extended connectivity fingerprints. However, since the number of available training experiences is limited due to the high cost of experimentally deriving the true labels, transfer and multitask learning strategies are usually incorporated in addition.

Virtual Screening is a brute-force approach to molecule design in which huge libraries are systematically scanned for molecules with specific target properties. Clyde [Cly22] gives an overview of how fast prediction models based on deep learning enable ultrahigh-throughput virtual screening (uHTVS) for the task of protein-ligand docking. A ligand is a molecule able to form a strong bond to a biomolecule, e.g., a protein. Identifying well-docking ligands is a fundamental concept of the drug design process. Virtual screening is further divided into ligand-based, structure-based, and hybrid methods. Ligand-based screening focuses on comparing candidate molecules to a set of known well-performing ligands. Consequentially, this form of screening requires a reliable means of comparing molecules with respect to certain target characteristics, e.g., the occurrence of specific pharmacophores or the match of molecular properties. Structure-based screening aims to identify promising candidates by directly estimating the binding energy between the candidate and the target protein. Finally, hybrid methods combine both strategies into one approach.

Since molecule libraries usually contain many potential structures and traditional docking tools are computationally expensive, only a small part of the molecule space can be covered with virtual screening. Machine learning models can be incorporated into this process as surrogate models to accelerate the analysis of each molecule. One possible strategy is to directly train a surrogate that predicts a docking score based on a molecular descriptor, bypassing the need for a docking program. However, in contrast to executing the docking program, this approach does not provide additional information, like the best-found docking pose. This may restrict the possibilities of subsequent expert analysis. An alternative approach is to use a machine learning model to filter the libraries for promising candidates and perform the complex docking program only for these molecules.

Chemical Synthesis. Even the search for a suitable candidate molecule for the task at hand is a complex undertaking. However, the results are of no use without a strategy to transform the theoretically well-functioning molecules into experimentally accessible ones [Tho+22]. Consequently, chemical synthesis is an important step in drug development and indispensable for validating molecules [Tho+22]. Chemical synthesis is one of the most time-consuming steps of the DMTA cycle, taking between 8–12 weeks [Tho+22]. One of the reasons is that the concrete synthesis path from a set of starting points to the final target molecule is rarely known. Therefore, drug design is in need of computer-aided tools able to derive a synthesis path automatically. Since this process starts from a target molecule and traverses the synthesis path backward until it reaches a set of commercially available starting structures, it is called

retrosynthesis [Tho+22]. Complicating matters further, the optimal synthesis strategy can dynamically change based on the availability of the starting material [Tho+22].

Historically, computer-aided synthesis planning was dominated by rule-based strategies that use a set of hand-coded transformation rules [Tho+22]. In recent years, however, more and more machine-learning-based approaches for retrosynthesis have been reported in the literature (e.g., [SPW18]). The process is fueled by huge datasets of chemical reactions mined from literature and patents. The models are either trained to predict the forward direction, i.e., predicting the product of the reaction between a set of reactants, or they are trained for backward reaction prediction, making them applicable for retrosynthesis. Training these models can be characterized as a supervised learning task since the expected results for a given set of inputs are known. However, synthesis planning poses some unique additional challenges compared to traditional supervised learning tasks. First, the output representation is more complex than the usual output in the form of a label, and second, the reaction data from the literature contains mostly positive examples, preventing the model from experiencing examples of reactions that do not occur [Tho+22].

All in all, there has been significant progress in the field of computer-aided synthesis planning in recent years. However, the widespread use of such techniques is still hindered by the sparse, noisy, and unstructured training data. It is, therefore, unlikely that this technology will render expert knowledge obsolete, even though it could become an indispensable tool for the latter in the future [Tho+22].

De Novo Molecule Generation. Finding a suitable molecule for a respective task is still dominated by combinatorial and high-throughput methodology. Although these strategies have undeniably led to success and AI can facilitate the process as discussed in the previous paragraphs, relying solely on information about known molecules restricts the search to a tiny section of the vast space of potential molecules. Discovering new molecules and bringing them to the market remains an extraordinarily expensive task. In drug development, this is even more complicated, as time-consuming clinical trials are required. De novo molecule design, i.e., the generation of novel molecules with specific property profiles, offers an alternative to the current trial-and-error approaches. It is not limited to reusing information about known structures but can freely explore chemical space and generate molecules not found in any chemical databases [DSC15].

Up to date, the DMTA cycle is the standard procedure for discovering new molecules, as discussed in Chapter 1. An AI-based generative model can function as an inner loop in the DMTA cycle. It can generate new, viable, and promising candidate molecules as a starting point, bringing innovation and potentially accelerating the process [Tho+22]. Furthermore, the generated molecules are created from scratch and tailored to the specific requirements of the problem. Consequently, generative models are an active area of research in molecular design [Tho+22].

De novo molecule generation has to deal with three major challenges [SC19]: How are molecules generated, how are molecules evaluated, and how are molecules optimized toward

the target properties? Throughout the research on this topic, various AI methods have been used to address these problems. The first approaches featured classical search algorithms such as depth/breath-first searches, evolutionary algorithms (EAs), and Monte Carlo sampling [Tho+22]. To date, there are several approaches in research based on the latter, which we will focus on in Section 4.3. In addition, with the rise of deep learning, deep generative learning, introduced in Section 2.3, has also made its way into molecule design, taking advantage of the vast amount of publicly available molecule data.

Devi, Sathya, and Coumar [DSC15] provide an overview of the role of EAs in de novo drug design. EAs are a well-researched tool for molecule generation, not least through their ability to optimize multiple objectives simultaneously. These approaches can be divided based on the way the algorithm constructs molecules. Some use atoms as their smallest building block, which lets them freely explore the whole space of potential molecules. However, they have to deal with a vast search space, and the generated molecules are not necessarily valid and synthesizable. Fragment-based approaches use a set of molecule fragments and combine them to generate new structures. This limits the search space and increases the frequency of realistic molecular structures, but on the other hand, reduces the flexibility of the search. In the case of drug design, both representation strategies can be further subdivided into structure-based and ligand-based approaches. Structure-based design centers the molecule search around a target binding site of a protein. Information about the structure of the binding site is directly incorporated into the search process. Likewise, the quality of the molecules generated is determined by the interaction energy with the target protein. The ligand-based design focuses on the similarity of the generated molecules to known ligands.

Finding an optimal drug candidate involves balancing multiple, often conflicting, objectives. This can include biological activity, oral bioavailability, synthesizing feasibility and many more [DSC15]. EAs can consider multiple objectives in their search by combining them into a single weighted objective. Alternatively, multi-objective EAs can develop a set of Pareto-optimal molecules, i.e., molecules for which no other molecule is better in all objectives [DSC15]. An overview of EAs and how they can be utilized for the multi-objective design of SARS-CoV-2 inhibitor candidates is further discussed in Chapter 4.

Utilizing deep neural networks for molecule generation is a relatively recent development, starting around 2016/2017 [Tho+22]. However, it has quickly attracted most of the community's attention because of the unique generation paradigm neural networks offer [Tho+22]. One of the significant advantages is that neural networks can benefit from the large amount of molecule data available. A deep generative model trained on this data can implicitly learn the underlying chemical principles to approximate the distribution that generated these molecules [Tho+22]. A generative model trained in such a fashion can be used to sample new but realistic molecules, like sampling a number from a random number generator [Pol+20]. Furthermore, such models can be trained to generate molecules conditioned on a specific target property, which enables efficient exploration of the search space [Pol+20]. The process of a generative model allowing traversal of the molecule space is pictured in Figure 3.4. In this example, the molecule generation model is trained to generate molecules from a two-

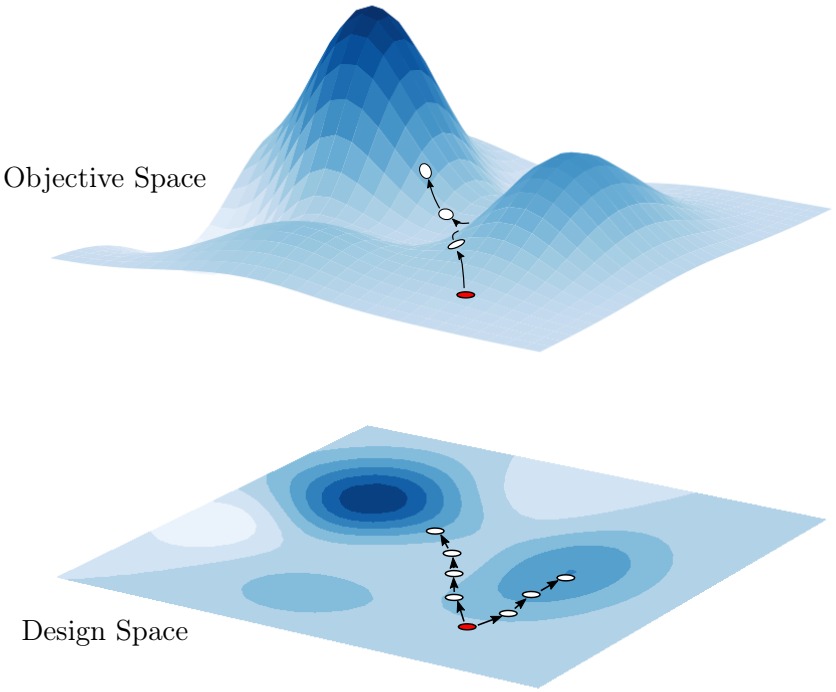


Figure 3.4: Navigation in the multimodal molecule landscape with the help of a deep generative model. This model embeds molecules in the two-dimensional design space. Starting from an origin molecule (red circle), the model can be used to explore the objective space by sampling similar but new molecules in the design space.

dimensional latent space, pictured at the bottom. The model can be used to sample new molecules of a certain distance to known solutions to explore the respective objective space of the target property, illustrated above. Many approaches utilize an additional property prediction network to evaluate the molecules with respect to these target properties. This setup enables a structured search in chemical space for promising molecular candidates.

Two questions are of essential importance when designing a deep generative model, which are both investigated extensively in research [SG20]: Firstly, on which architecture is the model based in order to allow efficient training and sampling? Secondly, how can molecules be represented in a computer-friendly way? Various studies have proposed different approaches based on standard deep generative models, such as variational autoencoders, recurrent neural networks, and generative adversarial networks. Likewise, there are multiple possible representations a generative model can utilize, as listed in Section 3.2. A more in-depth discussion of how molecules can be represented for deep generative design and an overview of the state-of-the-art techniques are provided in Chapter 5 and Chapter 6.

As with generative models in general, it is difficult to evaluate and compare the performance of different molecule generation models. Benchmarking frameworks like molecular sets

(MOSES) [Pol+20] and GuacaMol [Bro+19] try to provide a standardized suite for comparison. They offer training, test, and validation datasets and a collection of metrics to evaluate the generated molecules. One of the most important qualities of a generative model is the ability to generate valid, unique, and novel molecules. A valid molecule does not violate chemical constraints, e.g., atom valences. Uniqueness is usually measured by generating a fixed number of molecules and calculating the ratio of uniquely occurring samples. This metric is the first indicator of the model's capability to generate a diverse set of samples. All unique molecules not already contained in the training data are considered novel. Generally, a good generative model is able to generate a variety of novel molecules. In addition, the molecules should also resemble the properties of the training data in such a way that they could realistically have originated from the same ground-truth distribution. The mentioned benchmarking frameworks offer a variety of similarity and diversity measures to evaluate this property. These metrics represent good properties that every molecular generation model should have. Furthermore, generative models can be conditioned on additional molecule properties to allow a goal-directed de novo molecule generation. For example, in drug design, the generated molecules can be evaluated for their biological activity, synthetic feasibility, similarity to known drugs, or non-toxicity.

3.4 The Impact of De Novo Molecule Design

The previous section highlighted the various application possibilities for AI in computer-aided molecule design. Especially de novo design has the potential to accelerate the molecule design process with innovative new solutions. But are these theoretically well-performing candidates transferable to real-world problems? This section briefly reviews the applications of de novo molecular design, its potential, and the challenges associated with this new design philosophy.

In 2019, Schneider and Clark [SC19] provided an overview of recent de novo molecule design applications. The review is focused on classical optimization approaches from AI rather than deep generative models, as there are fewer application studies for this newer family of algorithms. The listed studies include approaches for the design of various inhibitors, e.g., for β-Secretase, Aurora Kinase A, Janus Kinase 3, and Helicobacter Pylori HtrA. Furthermore, some studies investigated design problems with multiple objectives, like the generation of selective matrix metalloproteinase 2 (MMP-2) inhibitors. This study aimed to generate novel inhibitors for MMP-2 that show selectivity over MMP-9. A further example is the design of dual-target inhibitors for COX-2 and LTA_4H.

Overall, the authors note that the investigated algorithms have the potential to generate synthesizable molecules with the desired biological activity. However, in a majority of the cases, the molecules did not show the same performance as they did in silico. Considerable manual effort and further refinement were often required, such as modifying the molecules to facilitate their synthesizability. These results are not necessarily a failure but only highlight the fact that the algorithms studied are only part of the overall molecular design process. De novo design programs can play a useful role by generating innovative starting points and

novel ideas. For example, algorithms can suggest molecules as starting points for hit-to-lead chemistry, where they are on par with other traditional hit-finding methods, such as viral screening.

All in all, Schneider and Clark [SC19] note that modern design tools are, to a certain degree, able to generate synthesizable molecules with the desired property profile. Although de novo design can not propose a perfect compound in a one-shot fashion, it can provide high-quality ideas for a subsequent investigation by experts. Scoring the molecules remains an Achilles heel of the investigated approaches. Moreover, the search is complicated by the highly nonlinear relationships among the many factors determining a drug's pharmacological effect.

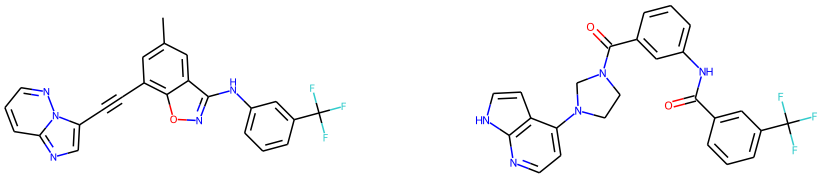


Figure 3.5: The two most promising DDR1 inhibitors found by Zhavoronkov et al. [Zha+19] with the help of deep learning.

A prominent example of the application of deep learning to drug design is the work of Zhavoronkov et al. [Zha+19] on the design of discoidin domain receptor 1 (DDR1). They introduced a generative tensorial reinforcement learning algorithm (GENTRL), combining a variational autoencoder trained on the ZINC database with subsequent reinforcement learning. After initial training, the autoencoder was fine-tuned on known DDR1 inhibitors. Reinforcement learning was used to generate potential inhibitors with respect to three reward functions. The reward functions were realized utilizing self-organizing maps (SOMs) and favored molecules that inhibited DDR1, were specific for DDR1 and had some degree of novelty.

The authors were able to select a set of six promising candidate molecules after only 23 days, which were successfully synthesized after a further 12 days. The lead candidates were tested in vitro for their inhibitory activity. Two of the six compounds, pictured in Figure 3.5, strongly inhibited DDR1 while at the same time showing excellent selectivity over DDR2. As the authors note, the proposed molecules could benefit from further optimization regarding selectivity, specificity, and other medicinal chemistry properties. Although the study was followed by some controversy regarding the similarity of the main candidate compound with the drug ponatinib, the results impressively demonstrate the potential of deep learning in drug design [MRD22].

In their review on the importance of AI in drug design, Schneider et al. [Sch+20] hypothesize that with increasing accuracy of prediction models, the whole DMTA cycle could at some point become virtual. The actual synthesis would only be an intermediate step to ensure that

the process is proceeding in the right direction. Hypothesis generation could become significantly faster while proposing more promising molecules, including the particular synthetic path. Ultimately, such tools could support chemists to become more effective and drastically reduce the duration of DMTA cycles. However, one of the most significant challenges in combining AI and drug discovery is the appropriate mindset and discovery "culture" of all parties involved. The authors advocate that all stakeholders recognize and acknowledge people's different expertise and must collaborate to develop common terminology and methodology. Crucial requirements for such cooperation are the availability and security of data, robust algorithms, and modular platform pipelines. It should be noted that the goal of AI-driven drug design is not to replace chemists and designers but rather to augment them. However, drug discovery scientists must accept the value AI for this to happen. Interaction between scientists and an adaptive AI might be crucial to overcoming the complex challenges of drug design. This thesis is supported by Schneider and Clark [SC19], who note that chemists and computer scientists often seem disconnected. Healthy skepticism about new data-driven technologies is sometimes overshadowed by simple denial of their potential among some medical chemists. However, such tools should not be seen as a threat to one's expertise but should be integrated into a collaborative workflow that leverages the respective strengths. The promise of accelerated, personalized healthcare that meets the challenges of the 21st century and provides highly targeted and effective treatments for everyone should be well worth the effort.

3.5 Ethical Implications

As outlined in the previous sections, the application of AI in drug discovery holds incredible potential for personalized, rapid, and affordable healthcare. Moreover, healthcare is just one of the many areas of our daily lives that benefit from molecular design tailored to the problem. Overall, the technology has enormous potential to bring significant advances to humanity. AI-guided molecule design is already a powerful tool that is continuously improved year after year. While continuous improvement and application are important, discussing the ethical implications of such an influential technology should not be ignored. This section points out some ethical questions that arise with the advancement of AI in molecule design, intending to raise awareness for this topic.

In a thought-provoking study, Urbina, Lentzos, Invernizzi, and Ekins [Urb+22] demonstrated how a molecule generation model could potentially be misused. They work for a company called Collaborations Pharmaceuticals, Inc., which owns a generative model with an extension to predict toxicity. This toxicity model usually penalizes molecules that could be harmful to humans in order to guide the molecular design in the right direction. Originating from a simple thought experiment, the authors carried out a proof-of-work study and inverted this objective, resulting in the generation of toxic molecules. In addition, the search was guided toward compounds similar to the nerve agent VX, a highly toxic chemical warfare agent. With this setup, the authors were able to generate 40 000 new molecules within the

desired objective thresholds in only six hours. Many generated molecules looked plausible but were predicted to be even more toxic than the original nerve agent.

None of the molecules found were actually physically synthesized, nor was any molecule analyzed for synthesis capability or retrosynthesis software used. However, many software components similar to those used in the generation process are readily available as open-source software. In addition, there is a range of commercial companies offering chemical synthesis.

The authors conclude that dual use of AI guided drug discovery is possible. With their proof-of-work study, they wanted to raise awareness of this possibility and encourage discussion of such issues among the community of researchers and practitioners. However, the question arises: "Can we lock away all the tools and throw away the key?" [Urb+22]. The authors hypothesize that further means of restricting access to such technologies might be needed like it is done for GPT-3 in the domain of language modeling. Further, common codes of conduct for responsible science, such as The Hague Ethics Guidelines, could motivate companies to raise awareness among their employees, protect the technology, and prevent misuse. Universities, on the other hand, should also emphasize the ethical training of students and broaden the scope to other disciplines so that the potential misuse of these technologies is discussed from the very beginning.

Another problem regarding AI in drug development is whether the invented molecules are actually patentable [Heu18; KC18]. The development of a new drug usually costs a considerable amount of money. In order to make the process economically feasible, the invented compound must be protected by a global patent. In the case of a drug designed by AI, it is, however, debatable whether it was invented by a person. Under U.S. law, for example, only people are entitled to patents. But if a designer can not explain how the AI derived a particular compound, can she or he still be seen as the inventor? Patent law will likely have to adapt to the change in drug development brought about by AI.

A further issue posed by data-driven AI technologies could be the reproduction of biases embedded in the training data. The quality of the training data is a key factor for the quality of the generative model. In the case of a systematic bias in the training data, the model is most likely unable to represent the true observation space fully. An example of such a bias in medicine is gender bias [Ham08]. For a long time, clinical trials have been performed on populations consisting mainly of young or middle-aged white men. Although the situation has improved in recent years, such gender or race biases could influence the molecules used in the AI-driven design process, and the resulting model could reproduce the biases. Researchers should be aware of these issues and make special efforts to ensure that the data sets used are inclusive and balanced.

Part II

Molecule Design

4 Evolutionary Multi-Objective Design of SARS-CoV-2 Protease Inhibitor Candidates

Designing and testing molecular structures suitable for a given problem can be a challenging and costly task prone to failure [Bro+20a]. Chemical structures are complex, the space of potential candidate molecules is vast, and experiments and trials can be time-consuming. Incorporating AI methods in this process has the potential of developing new compounds designed according to desired properties, and the respective problem requirements [Sch18]. In the following, we introduce an approach based on a GA capable of proposing candidate molecules for protease inhibition and demonstrate its application to the design of drug candidates for the 2019 novel coronavirus (SARS-CoV-2). Inhibiting the functioning of the virus protease is a possible way of limiting its replication. This can be achieved by identifying a suitable biomolecule (the ligand) that can bind with the virus protease enzyme and thereby inhibit its function. The crystal structure of the SARS-CoV-2 main protease M^{pro} is known [Jin+20] and, therefore, finding a ligand expressing a high binding affinity to the protease poses an optimization problem. However, the binding affinity is only one of several objectives that can be considered in the search for a suitable drug candidate, like its drug-likeness or synthesizability. The approach presented hereinafter takes multiple such molecular metrics into account and utilizes a multi-objective GA to propose candidate ligands. It is based on the published article: "Evolutionary Multi-objective Design of SARS-CoV-2 Protease Inhibitor Candidates" [Cof+20].

Outline. Section 4.1 introduces the basic concepts of GAs and how this family of algorithms represents, modifies, and evaluates solutions. In Section 4.3, an overview is given on the application of GAs in molecule optimization and generation. Finally, in Section 4.4, we present an approach on how GAs can be utilized for the multi-objective design of SARS-CoV-2 protease inhibitor candidates.

4.1 Genetic Algorithms

GAs are considered part of the collective term EAs describing a family of optimization techniques. The following summary is an adaption of the overview given by Rudolph and Schwefel [RS94].

EAs are inspired by the evolutionary process occurring in nature. In the course of time, evolution has created life forms that are almost optimally adapted to their respective habitats. The underlying principles inspired a family of algorithms, commonly combined under the name of evolutionary algorithms. First occurrences date back to the early sixties as Schwefel [Sch77] and Rechenberg [Rec73] introduced the evolutionary strategy, Holland [Hol75] developed the GAs, and Fogel, Owens, and Walsh [FOW66] introduced evolutionary programming. Throughout this paper, we will focus mainly on the subclass of GAs, which will be discussed in more detail below.

GAs are commonly applied to optimization problems to approximate optimal solutions. The concept of optimization as one field of artificial intelligence has already been described in Section 2.1. A core concept of optimization problems is the fitness function $f : X \to \mathbb{R}$ mapping a solution $x \in X$ to its fitness value, with X being the space of possible solutions. The algorithm strives to find a solution that approximates the optimization objective

$$\min_{x \in X} f(x) \tag{4.1}$$

One of the main advantages of GAs is that these algorithms can be applied for black-box optimization, as they iteratively and stochastically generate and improve solutions for the given problem until a satisfaction criterion is met [Mir19]. This makes them applicable for multimodal optimization problems or problems containing nonlinearity and discontinuity [RS94].

Inspired by nature, a single solution for a given problem is commonly referred to as an individual. GAs simultaneously process a set of individuals, the so-called population. Therefore, the algorithms can consider different optimization paths at the same time, preventing premature stagnation and making GAs scale well with more computational resources [RS94]. Algorithm 1 pictures a common structure of GAs and is inspired by Rudolph and Schwefel [RS94]. At the first time step $t = 0$, a set of μ individuals is initialized to form the population P. Individuals can be initialized completely randomly or with variations of already known solutions. After initialization, all individuals in the population have to be evaluated regarding the respective fitness function. This step is followed by a cycle of so-called generations that is repeated until a chosen termination criterion is met. For every generation, a set of child individuals is created by recombining individuals from the population. After its creation, a child can be randomly altered by mutations to increase the diversity in the population. Finally, the newly generated individual is evaluated with the fitness function. The new population is then composed by selecting the μ best performing individuals from the current population and the children. This selection of only well-performing individuals lets GAs converge toward optimal solutions.

The recombination and mutation functions are usually referred to as genetic operators. Over the years, various possible operators have been introduced in the literature. Usually, recombination functions as a way of mixing traits of multiple individuals into one child. Because the algorithm favors high-performing individuals for recombination during selection, the child has the opportunity to inherit well-adapted low-order schemas that collectively lead

Algorithm 1 General structure of GAs.

1: $t \leftarrow 0$
2: initialize P_0 with individuals $x_1, \ldots, x_\mu$
3: evaluate $f(x)$ for every $x \in P_0$
4: **repeat**
5: $C \leftarrow$ recombine individuals from P_t
6: mutate individuals in C
7: evaluate $f(x)$ for every $x \in C$
8: $P_{t+1} \leftarrow$ select fittest individuals from $P_t \cup C$
9: $t \leftarrow t + 1$
10: **until** termination criterion is met

to even better adaptation. Mutation operators usually randomly change minor aspects of an individual, allowing for small adaptations over the course of evolution. The concrete implementation of these operators usually depends on how the individuals are represented. Common representations include sequences of bits, integers, or floats, but also more complex data structures like graphs.

4.2 Virus Protease Inhibition

In the work presented hereinafter, a GA is utilized to design protease inhibitor candidates. A protease inhibitor is a biomolecule able to bind to a protease, preventing it from performing its normal function. Binding refers to the ligand positioning itself in a target binding site—the so-called pocket—and forming various non-covalent interactions like hydrophobic interactions, hydrogen bonding, π-stacking, salt bridges, and amide stacking [FS17]. A possible application of protease inhibitors is as antiviral drugs.

In the following work, we will demonstrate the AI-guided design of molecules that could function as such antiviral drugs. The entire process is illustrated by the development of SARS-CoV-2 inhibitors but can be easily adapted to the development of inhibitors for any known proteases. The disease COVID-19 is caused by SARS-CoV-2, an RNA virus from the family of coronaviruses. It replicates by entering the cells of a host and taking over the cell's replication mechanism. A critical step of the replication is the cleavage process. Precursor polyproteins have to be cut into mature non-structural proteins by the virus protease. A ligand that binds to this protease can potentially inhibit this cleaving process and interfere with the reproduction, as schematically pictured in Figure 4.1.

The described binding process can be computationally modeled. These models can be utilized to estimate how well a ligand binds to a target protein. However, various factors affect the concrete binding affinity, like the protein-ligand geometry, chemical interactions, and various constraints and properties like hydration and quantum effects. Molecular dynamics computations can be used to estimate the binding affinity but are complex and expensive. Docking tools, like the software AutoDock [Mor+09], use heuristics and simplifications of the physical reality to estimate the binding affinity. These tools aim to provide a sufficiently accurate measurement while reducing the computational cost of the calculation.

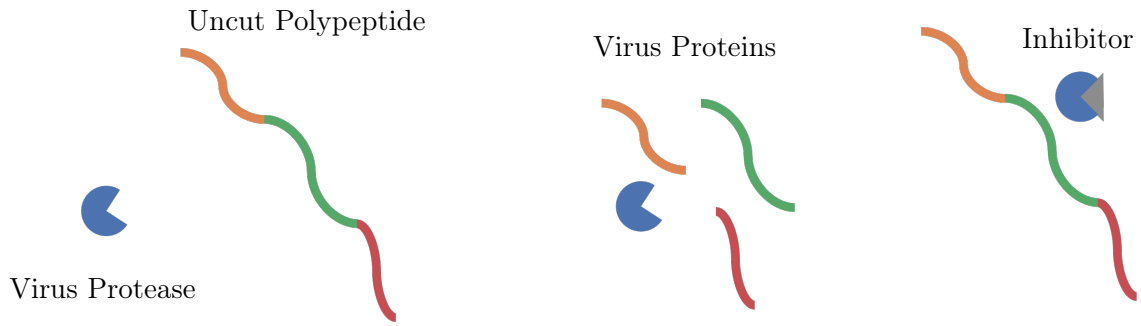


Figure 4.1: An illustration of the cleavage process. A protease enzyme cleaves the precursor polyproteins into virus proteins. An inhibitor molecule can prevent the process.

Binding affinity calculations require information about the target structure, and for SARS-CoV-2, this structure is known, e.g., [Dai+20; Jin+20; Zha+20]. This has led to the design and proposition of various inhibitor candidates. Some proposed drugs are based on inhibitors for other viruses [Cal+20]. Others are found by virtual screening of molecule libraries [Fis+20] and by computational drug design [MPP20]. Although computational drug design can facilitate the search for a suitable drug candidate and can propose promising new molecules, it has to be noted that such new molecules require extensive testing. However, these methods have the potential to identify innovative starting points in the search for a potent inhibitor.

4.3 Related Work: Genetic Algorithms and Molecule Design

Several adaptations of AI for the de novo drug design can be found in literature [DSC15; Bro+19]. The methods vary in their pursued objective; some target finding drug candidates for a specific binding site [PHK01; YPL20], whilst others focus on generating drug-like molecules in general [DTG00; Pol+20]. Furthermore, the proposed methods differ in how they construct molecular structures. Some approaches work directly on an atom level, constructing molecules atom by atom and bond by bond [DTG00; Nig+20]. Other approaches use sets of chemical fragments and represent molecules as compositions of such fragments [PHK01].

There are various examples of GAs being utilized for molecule optimization and generation. For example, single-target GAs have been successfully applied to the optimization of peptide ligands [RBF12; Kra+18] by incorporating feedback from in-vitro experiments. Douguet, Thoreau, and Grassy [DTG00] demonstrated how a GA can facilitate the process of finding drug-like molecules. Therefore, they utilized the SMILES representation and altered molecules directly on an atomic level. In contrast to the work presented in the following, their algorithm is not targeted at a specific ligand. Although their approach also takes multiple objectives into account, they are simply combined into a weighted sum and optimized as a single objective. A further example of GAs operating on the SMILES representation is the work by Nigam, Friederich, Krenn, and Aspuru-Guzik [Nig+20]. Their approach can be applied to various molecule design tasks and incorporates deep neural networks. The networks are included as a means to increase molecule diversity, acting as an adaptive fitness function by penalizing

long-surviving molecules. The Molecule Evoluator by Lameijer, Kok, Bäck, and IJzerman [Lam+06] is a general-purpose tool for the evolution of drug-like molecules. This program works with a user-defined fitness function and also constructs molecules on an atomic level.

The approaches presented so far are not specific to protein inhibition but are general methods for de novo drug design. With ADAPT, Pegg, Haresco, and Kuntz [PHK01] presented a fragment-based GA for molecule design able to optimize molecules to fit a specific binding site. The approach represents molecules as acyclic graphs of molecular substructures and evaluates a molecule's fitness based on docking simulations with the target protein. A further example of a fragment-based evolutionary de novo drug design tool is LigBuilder by Yuan, Pei, and Lai [YPL20]. This software suite can be used to analyze a given binding site and offers different strategies for constructing fitting ligands.

Finally, there are some examples of studies treating molecule generation as a multi-objective optimization problem. Brown, McKay, Gilardoni, and Gasteiger [Bro+04] introduced a multi-objective GA for molecules in a graph representation that features a Pareto ranking scheme. In the approach for the identification of central nervous system drugs presented by Wager, Hou, Verhoest, and Villalobos [Wag+16], six physicochemical properties are jointly analyzed. However, unlike the approach presented here, this tool is based on medical knowledge rather than evolutionary algorithms. A further multi-objective GA has been proposed by Horst et al. [Hor+12]. Their approach targets the design of adenosine receptor ligands and includes support vector machines. An overview of multi-objective optimization for drugs, with the underlying problem definitions and different optimization methods, is given by Nicolaou and Brown [NB13].

4.4 Evolutionary Multi-Objective Molecule Search

In the following, we present an evolutionary multi-objective approach for molecule design and apply it to the design of SARS-CoV-2 protease inhibitor candidates. As mentioned in Section 4.1, a core concept of GAs is the fitness function. Therefore, Section 4.4.1 provides an overview of how molecule candidates are evaluated in the presented approach. Since we treat molecule generation as a multi-objective problem, we introduce multiple molecule metrics to the evolutionary process. Furthermore, the GA requires a molecular representation for its individuals, which is introduced in Section 4.4.2, and operators to generate new molecules, which are introduced in Section 4.4.3.

4.4.1 Molecule Metrics

We employ five molecule metrics to evaluate the inhibitor candidates. Table 4.1 provides an overview of these metrics and their value ranges. Since all values differ in their range and optimum, we scale the metrics to a standardized range of $[0, 1]$ with 0 being the optimum. Since the binding affinity has no upper or lower limit, the metric is rescaled with regard to a maximum binding energy of $1\,\text{kcal/mol}$ and a minimum of $-15\,\text{kcal/mol}$. Rescaling is done with the help of soft clipping [KP20]. All metrics are briefly described in the following.

Table 4.1: Value ranges and optima for the employed metrics [Cof+20].

	Docking score [kcal/mol]	SA	QED	NP	Filters
Value range	$\mathbb{R}$	$[1, 10]$	$[0, 1]$	$[-5, 5]$	$\{0, 1\}$
Optimum	$-\infty$	1	1	5	1

Binding Affinity Score. Since the main objective of this approach is the design of protease inhibitor candidates, we evaluate the binding affinity between the generated ligand and the respective binding site on the target protease. A commonly used tool for estimating the binding energy is the automated docking tool AutoDock [Mor+09]. To allow a fast execution, AutoDock embeds the molecule in a grid and utilizes grid-based lookup tables. These lookup tables are created by placing the probe atom sequentially at each grid point and calculating the binding energy utilizing semi-empirical force field methods. This grid-based approach allows for a fast evaluation of a conformations binding affinity and enables the application of a Lamarckian GA. The GA generates and optimizes a population of conformations to find the best conformation with the lowest binding energy. Furthermore, AutoDock allows modeling parts of the receptor as flexible, which can increase prediction accuracy.

Various improvements have been made to AutoDock resulting in different versions of the software. AutoDock Vina [TO09] exchanges the force field method used in AutoDock with a hybrid scoring function based on empirical and knowledge-based data. QuickVina [Han+12] and QuickVina 2 [Alh+15] modify the search algorithm to improve the computation time required. This is achieved by identifying promising ligand positions and only executing the complex calculations for those positions. Additionally, QuickVina 2 simplifies various physical properties, e.g., neglecting water molecules and electrical properties of the ligand and the protein that change due to their interaction.

Despite all simplifications, Gaillard [Gai18] demonstrated that AutoDock Vina provides a better binding affinity prediction when compared to various other computational docking methods. Alhossary, Handoko, Mu, and Kwoh [Alh+15] compared Quickvina 2 and Autodock Vina and found them to be comparable in accuracy. Our multi-objective GA assigns a fitness to every generated candidate molecule, and every fitness evaluation requires a binding energy calculation. We employ QuickVina 2 for these calculations, as it provides an acceptable balance between computation time and accuracy.

Synthetic Accessibility (SA). The main objective of the presented approach is to find an inhibitor candidate with a strong binding affinity to the target protein. However, a candidate is only useful in practice if it is also easy to synthesize. A complete retrosynthetic analysis—finding a synthesis path based on commercially available structures—is not feasible. Instead, we use the synthetic accessibility (SA) score proposed by Ertl and Schuffenhauer [ES09]. This heuristic metric estimates the synthetic accessibility of drug-like molecules on a continuous scale. The authors showed that the metric has a high degree of agreement with manual estimates from experts.

Quantitative Estimate of Drug-likeness (QED). In addition to ease of synthesis, a key challenge of the presented approach is generating molecules suitable as drug candidates. When considering if a molecule is drug-like, it is useful to investigate how similar it is to existing drugs. This comparison can be based on different molecule properties. A commonly used metric is the Lipinski rule of five [Lip+97] that considers the number of hydrogen bond donors and acceptors, the molecular weight, and the octanol-water partition coefficient (logP). The rule specifies value ranges for these metrics, and a molecule falling out of more than one of these ranges is not considered drug-like. However, incorporating Lipinski's rule of five into our approach would have drawbacks. First, the rule is a binary metric; it only expresses whether a molecule meets the criteria or not. Furthermore, there are examples of molecules among modern drugs that violate more than one of Lipinski's rules. Therefore, we employ the metric introduced by Bickerton, Paolini, Besnard, Muresan, and Hopkins [Bic+12], the quantitative estimate of drug-likeness (QED). The QED is a combination of multiple molecular properties. In contrast to Lipinski's rule of five that considers fixed value ranges for every property, QED has an individual desirability function per property. The desirability scores are combined in a geometric average to form a single drug-likeness score. Including QED into the GA's fitness evaluation should bias the generated molecules toward more realistic and drug-like molecules.

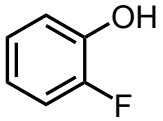
Natural Product-likeness (NP). To further incentivize the algorithm to generate realistic molecules, we include the natural product-likeness (NP) in the fitness evaluation. This score introduced by Ertl, Roggo, and Schuffenhauer [ERS08] is based on molecular properties that commonly differ between synthetic and natural molecules. Exemplary structural features considered by this score are the number of aromatic rings and the frequency of nitrogen and oxygen atoms. Nature has produced and evaluated numerous bioactive structures, and including such a metric in our approach should therefore increase the quality of the generated molecules.

Medical Chemical Filters. Considering the target of generating drug candidates, the GA should exclude potentially toxic molecules. In addition, the algorithm should also avoid generating unstable molecules with potentially toxic metabolites. Therefore, our approach considers the medical chemical filters (MCF) and PAINS filters described by Polykovskiy et al. [Pol+20]. The filters are fast to calculate and check important properties of drug candidates. Although the filters are a binary metric, we include them as an additional optimization objective rather than a hard constraint. Molecules violating the filter may still survive the selection if they perform well in other metrics, and these molecules could be helpful intermediate steps in the evolutionary process.

4.4.2 The SELFIES Representation

The previous section introduced the metrics used to analyze the generated molecule candidates. The question remains open, how the algorithm represents and evolves these molecules? Section 3.2 already provided an overview of different families of molecule representations.

In this approach, we utilize a string representation, but instead of opting for the commonly used SMILES, we employed the more recently introduced SELFIES representation [Kre+20]. SELFIES—like SMILES—is a text-based representation. Figure 4.2 provides an exemplary comparison of a molecule's structural formula and its SMILES and SELFIES representation.



SMILES: Oc1ccccc1F
SELFIES: [O][c][c][c][c][c][c][Ring1][Branch1_1][F]

Figure 4.2: Molecular structure formula, SMILES, and SELFIES of 2-fluorophenol [Cof+20].

SELFIES represents molecules as a sequence of tokens. Each token encodes a part of the molecular graph, e.g., the occurrence of an atom or the start of a branch or ring. It bears advantages over the SMILES that make it particularly suitable for the application in GAs. A key aspect of SELFIES is that it is a 100% robust representation, meaning all SELFIES encodes valid molecules and all molecules can be encoded as SELFIES [Kre+20]. This is achieved by decoding every token with respect to a set of derivation rules. These rules specify a formal grammar for the interpretation of SELFIES. For each token, there is a set of possible derivation rules. Which rule is chosen during decoding depends on the current state of derivation. The state of derivation keeps track of chemical constraints and ensures that only those rules are chosen, fulfilling the constraints. For example, the rules ensure that the maximum number of valence bonds is not exceeded. Since SELFIES always encode valid molecules, applying random alterations to them also results in a valid molecule. This benefit was already investigated by Krenn, Häse, Nigam, Friederich, and Aspuru-Guzik [Kre+20] and makes SELFIES especially applicable for GAs and deep generative models.

In our approach, each individual is represented by a SELFIES encoding one candidate molecule. The initial population is generated by randomly sampling fixed-length sequences of SELFIES tokens from a list of possible tokens. Since different SELFIES can encode the same molecule, every individual is translated to the canonical SMILES string and compared with a list of all previously generated molecules. If a molecule has already occurred, it is discarded, and a new candidate is initialized until the initial population is of the desired size. This process ensures that the algorithm starts with a population of unique molecules.

4.4.3 Genetic Operations

Genetic operators allow evolving new individuals from already well-performing ones. They are a central component of every GA, as they enable the traversal of the search space toward optimal solutions. Many common mutation operators, such as the biologically inspired point mutation, randomly alter an individual's genome with a certain probability, called the mutation rate. In our case, an individual's genome is defined as its sequence of SELFIES tokens, each token representing one gene. In the commonly used string-based SMILES representation, a random mutation operator would most likely generate a high fraction of invalid molecules. One of the reasons is that SMILES strings follow strict syntactic rules (e.g., closing parenthe-

ses), and a random SMILES string is, therefore, most likely syntactically invalid. Additionally, a SMILES string, although syntactically valid, may encode a molecule with incorrect atomic valences. The SELFIES representation, however, ensures that a string resulting from random mutations still encodes a valid molecule. This allows us to apply adaptations of common nature-inspired mutation operators that are described in the following:

Replacement exchanges a SELFIES token with a random one. Each token in the genome has an independent mutation rate of p_r, so multiple replacements are possible per genome during a mutation.

Insertion selects a random SELFIES token from the list of available symbols and inserts it at a random position in the individual's sequence of SELFIES tokens. This mutation is applied to an individual with a probability of p_i.

Deletion removes a random token from an individual's sequence of SELFIES tokens. This mutation is applied to an individual with a probability of p_d.

The set of available SELFIES tokens is inspired by those used by Krenn, Häse, Nigam, Friederich, and Aspuru-Guzik [Kre+20]. It contains tokens for the commonly occurring atom types $\{\mathrm{C}, \mathrm{O}, \mathrm{N}, \mathrm{F}, \mathrm{S}\}$ and for rings and branches of different types. We extended this list with benzene as a separate, composed token since benzene is a common substructure in organic molecules. Including it as an extra token should facilitate the generation of more complex molecules. Furthermore, every token is associated with an adjustable weight to control the probability of its selection during mutation. This weight can increase the probability of more common tokens, such as [C], in contrast to more complex ones, like ring structures.

4.4.4 Fitness Evaluation

The fitness evaluation is a core concept of GAs. By assigning a fitness to each individual, selection pressure can be applied to the population, driving evolution toward higher-performing individuals. Each individual is evaluated with a fitness function $f(x)$ that is based on the molecule metrics introduced in Section 4.4.1. All metrics are rescaled to a value range $[0, 1]$, with zero being the optimum.

A straightforward way of optimizing molecules with respect to the aforementioned metrics is to aggregate them into a single fitness score. This strategy will be featured as one experimental condition in the following and referred to as single-objective optimization. In this condition, an individual's fitness is determined by the weighted sum, which for a number of n metrics is defined as:

$$f(\mathbf{x}) = \sum_{i=1}^{n} w_i f_i(x) \tag{4.2}$$

We configure the weights to $\mathbf{w} = (0.4, 0.15, 0.15, 0.15, 0.15)$ with i corresponding to 1: docking, 2: SA, 3: QED, 4: NP, and 5: filters. These weights were chosen based on preliminary experiments. Since generating drug candidates is the main focus of this work, the docking

score is given the highest weight and, thus, the greatest influence on a molecule's fitness. All other metrics are weighted equally.

The fitness calculation for all individuals in a population is performed in parallel, making the algorithm scale well with more computational resources. The first step in calculating an individual's fitness is to convert its SELFIES to the SMILES representation. The software framework MOSES [Pol+20] is used to calculate the respective scores for QED, SA, NP and apply the MCF. The docking score is estimated using the software QuickVina 2, introduced in Section 4.4.1. Before a docking simulation can be performed with QuickVina 2, several preprocessing steps must be applied to the molecule representation. The cheminformatics software RDKit is used to estimate a conformation and convert the SMILES string to the three-dimensional PDB file format. The PDB file is converted to the PDBQT file format with the software MGLTools[1]. The PDBQT file is then used for the binding calculations in QuickVina 2. QuickVina 2 is also provided with a PDBQT file for the respective target receptor. In our case, the binding affinity is estimated for the COVID-19 M^{pro} (PDB ID: 6LU7 [Zha+20])[2]. However, an adaptation to other targets is easily conceivable by replacing the target file. To limit the binding calculations to the desired target area of the receptor, a search grid with a size of $22 \times 24 \times 22$ Å^3 is configured and centered around the position of the native ligand. QuickVina 2 offers an exhaustiveness parameter to balance between accuracy and execution time. We kept this parameter at its default value of eight for all experiments presented below. Overall, this configuration leads to an execution time of just a few minutes per molecule.

4.4.5 NSGA-II

In the previous section, we introduced the single-objective approach that defines fitness as a weighted sum of five molecular sub-objectives. However, when multiple objectives are involved, some are likely in conflict. It is, for example, conceivable that the GA tends to generate highly complex molecules with a strong binding affinity to the target. Such molecules would most likely be challenging to synthesize or unstable. Thus, the property profiles of the generated molecules strongly depend on the chosen weights of the metrics. Instead of combining all metrics into one fitness value, an alternative approach is to optimize all objectives jointly in a diverse population of molecules, each with a different tradeoff between these metrics. Therefore, in a second experimental condition, we replace the single-objective selection procedure with an explicit multi-objective approach called NSGA-II.

NSGA-II aims to evolve a population of non-dominated solutions, the so-called Pareto set. Formally, for molecules from the solution space $\mathcal{M}$ a Pareto set is described by $\{x^* \mid \nexists x \in \mathcal{M} : x \prec x^*\}$, where $x \prec x^*$ describes domination of an individual x over x^*. For a set of n different objectives with their fitness functions $f_1, \ldots, f_n$ domination is defined as $\forall i \in \{1, \ldots, n\} : f_i(x) \leq f_i(x^*)$, while $\exists i \in \{1, \ldots, n\} : f_i(x) < f_i(x^*)$. Therefore, the Pareto set consists of molecules that provide a good tradeoff between the different molecular metrics

[1] https://mgltools.scripps.edu
[2] PDB: protein database, https://www.rcsb.org

since, for each of these molecules, there is no other molecule that performs better on all metrics. The concrete fitness values of these solutions form the Pareto front in the objective space. NSGA-II aims to evolve a set of solutions with broad coverage of the Pareto front. In its first step, NSGA-II sorts the set of solutions based on their rank of domination. All non-dominated solutions are assigned the first rank and removed from the set. All remaining solutions that are no longer dominated form the second rank. This procedure continues until the original solution set is empty, and each solution is assigned a rank. The second step continues with all solutions from the first rank. The second or even higher ranks are appended if more solutions are needed. To construct the GAs next generation, NSGA-II chooses μ solutions from this selection that maximize a crowding distance. This results in the evolution of a set of solutions with broad coverage on the Pareto front.

Since our goal is to optimize molecules with respect to the five molecule metrics introduced in Section 4.4.1, the solutions are evaluated in a five-dimensional objective space. The MCF are a binary metric, and it would therefore be conceivable to include the satisfaction of these filters as a constraint rather than an additional objective. However, these filters are based on heuristics, and a molecule violating these filters could still prove to be a suitable drug candidate or potentially lead the optimization to promising areas of the search space.

4.5 Experiments

In this section, we evaluate the aforementioned GA for the design of SARS-CoV-2 protease inhibitor candidates. Furthermore, we compare the single-objective approach and the multi-objective approach based on NSGA-II. The configuration for these experiments is described in the following. We employed a $(10 + 100)$-GA design for the single-objective runs, i.e., the population size was ten, and 100 children were generated for every generation. Children were generated by selecting a random parent and applying the mutation operators described in Section 4.4.3, with mutation rates $p_r = 0.05$, $p_i = 0.1$, and $p_d = 0.1$. The next generation was composed of the ten fittest individuals selected from the parents and the children. The number of parent individuals was increased to 20 for the multi-objective runs. This should allow the NSGA-II to evolve a set of solutions with broader coverage of the objective space. All runs were performed for 200 generations, and we executed 20 independent runs per experimental condition. Based on the work of Krenn, Häse, Nigam, Friederich, and Aspuru-Guzik [Kre+20], we limited the maximum length of SELFIES tokens for individuals to 80 for all experiments.

4.5.1 Metric Development

Figure 4.3 shows the development of the five molecular metrics over the course of optimization. The graphs pictured on the left represent the scores of the best-performing individual after each generation, averaged over the 20 single-objective runs. The GA was able to continuously optimize the docking score with molecules passing the MCFs. However, the plots for QED, NP, and SA show that the GA struggled to improve these secondary metrics simultaneously. Especially after generation 140, the generated molecules seem to deteriorate in terms of SA

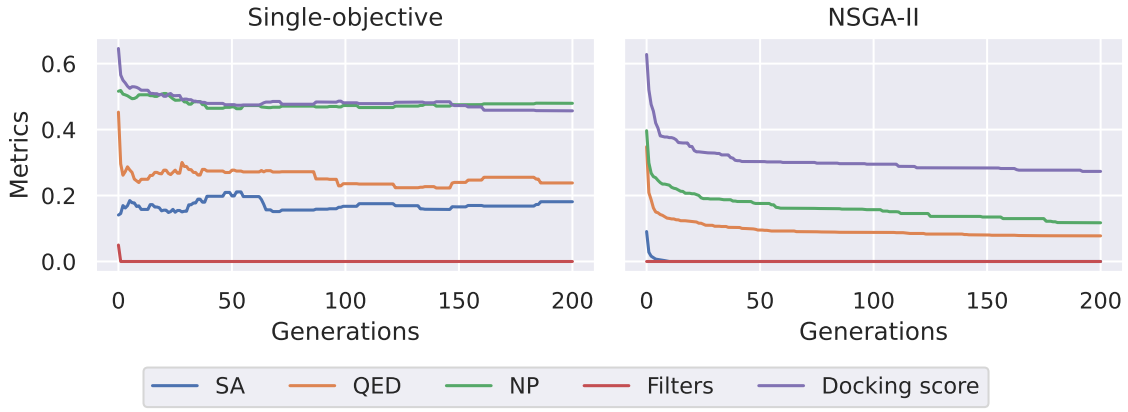
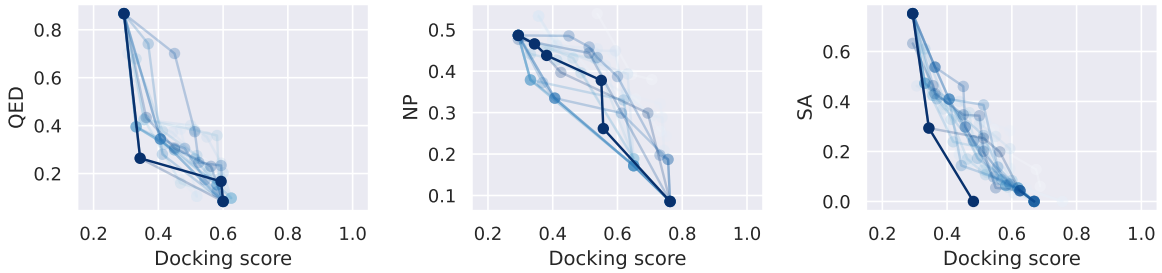


Figure 4.3: Development of the molecule metrics during (left) single-objective and (right) multi-objective NSGA-II optimization runs [Cof+20].

and QED in favor of a better docking score. This highlights how the different desired molecular properties conflict with each other. The algorithm has to balance between generating complex molecules tailored to the specific binding site and generating stable, synthesizable molecules with drug-like properties. Since the binding affinity had the highest weight, the algorithm concentrated on optimizing this property. In general, it is conceivable that the characteristics of the generated molecules strongly depend on the chosen fitness weighting.

The plots on the right describe the NSGA-II runs and show the best score achieved for each metric in each generation. Since the NSGA-II simultaneously optimizes every objective in the population, these results do not necessarily come from the same individual. As for the single-objective plots, the results are averaged over the 20 independent runs. The fitness plots show that the NSGA-II was able to steadily improve the population in the direction of well-performing individuals for each metric. However, an individual's improvement on one goal may lead to deterioration on other goals, which are not shown here.



(a) Docking score vs. QED (b) Docking score vs. NP (c) Docking score vs. SA

Figure 4.4: Typical Pareto fronts from an exemplary NSGA-II run: (a) docking score vs. QED, (b) docking score vs. NP, (c) docking score vs. SA [Cof+20].

To further analyze the NSGA-II runs and to verify whether the algorithm succeeded in generating molecules with a broad distribution of metrics at the Pareto front, we compare different two-dimensional slices of the evolved Pareto fronts in Figure 4.4. The plots picture the

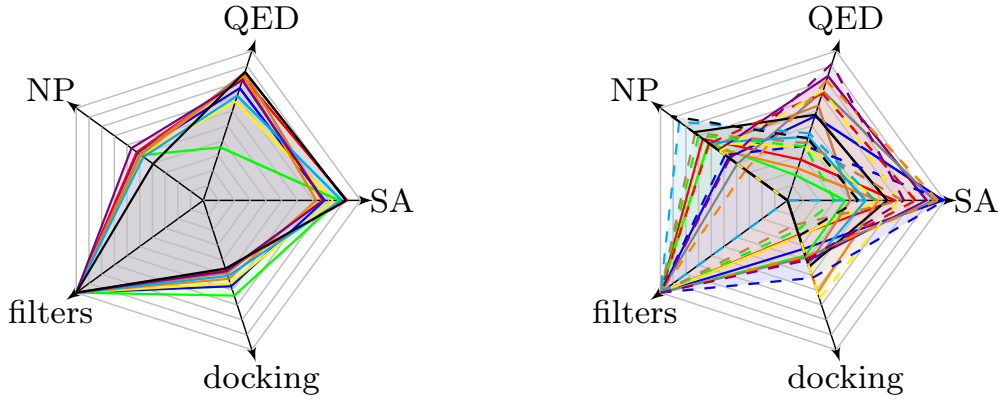
docking score and a respective secondary metric for the non-dominated individuals for every ten generations (earlier generations are marked with lighter shades of blue). The GA based on NSGA-II was able to generate molecules with varying degrees of balance between these metrics. Especially in Figure 4.4a and Figure 4.4c, it can be seen that NSGA-II has succeeded in evolving the set of non-dominant solutions toward the lower left corner. This tendency is not fully captured in the graphs presented, as they are only exemplary two-dimensional slices of the Pareto-front, and improvements in one metric can cause deterioration in another. However, the improvement of solutions is demonstrated by the hypervolume indicator, which on average improved from 0.10 ± 0.03 in the first to 0.20 ± 0.05 in the last generation.

Table 4.2: Experimental results of the best molecules generated by the single-objective approach and NSGA-II. All results are averaged over 20 independent runs. The values in the single-objective condition originate from the best individual found during evolution. The NSGA-II results correspond to the best objective found and do not necessarily originate from the same molecule. ▼ marks a minimization objective, while ▲ marks a maximization objective. Values for the native ligand N3 (from PDB 6LU7) and Lopinavir (a prominent drug candidate) are presented for comparison [Cof+20].

	Single-objective		NSGA-II		N3	Lopinavir
Objective	Best	Avg±Std	Best	Avg±Std	Value	Value
Fitness ▼	0.30	0.32±0.01	0.31	0.39±0.06	0.43	0.41
Docking score ▼	−9.30	−7.68±0.90	−13.30	−10.63±1.18	−8.40	−8.40
SA ▼	3.04	2.63±0.59	1.00	1.00±0.00	4.29	3.90
QED ▲	0.66	0.76±0.10	0.94	0.92±0.01	0.12	0.20
NP ▲	0.33	0.20±0.54	4.27	3.82±0.24	−0.18	−0.04
Filters ▲	1.00	1.00±0.00	1.00	1.00±0.00	1.00	1.00

An overview of final scores for the two experimental conditions is given in Table 4.2. For comparison, we also evaluated the five molecule metrics for the native ligand N3 from the PDB and an HIV main protease inhibitor; Lopinavir [KH05]. At the time the study was conducted, Lopinarvir was under consideration as a candidate SARS-CoV-2 inhibitor [MP20]. The results for the single-objective condition correspond to the score of the best-performing individual from the last generation. For NSGA-II, the results show the best score achieved in each metric, i.e., the scores mark corner points of the evolved Pareto front and do not necessarily come from the same individual. In our experiment, Lopinavir and the N3 bind similarly strongly to M^{pro}. With a docking score of $-9.3\,\mathrm{kcal\,mol^{-1}}$, the best molecule found in the single-objective condition surpasses these molecules while having a better estimated SA and scores better in terms of QED and NP. The NSGA-II was able to generate a well-performing molecule for every metric. On average, the best performing molecules in terms of SA, QED, and NP have near-optimal values in the respective objective. In addition to the outstanding best docking score of $-13.30\,\mathrm{kcal\,mol^{-1}}$, these results indicate that the NSGA-II can cover a wide area of the objective space. Obviously, these high values mark corner points of the Pareto front, and the corresponding molecules might be impractical. For example, the molecule with the docking score of $-13.30\,\mathrm{kcal\,mol^{-1}}$ showed unrealistic chemical structures.

Nevertheless, the results highlight the approach's capability of terminating with a diverse set of molecules. Therefore, NSGA-II enables a practitioner to choose from various solutions with varying degrees of balance between the objectives. Figure 4.5 underlines this advantage by comparing the molecules from the last generation of an exemplary single-objective and an NSGA-II run. The individuals from the single-objective condition have a similar fitness distribution, which might be caused by the fixed weighting of the molecular metrics. The only clearly differing molecule (marked in green) expressed a significantly worse QED in exchange for a slightly better docking score. The final population of the NSGA-II, on the other hand, consists of molecules with different fitness profiles. The plots vividly illustrate how the different objectives conflict and how no global optimal solution exists.

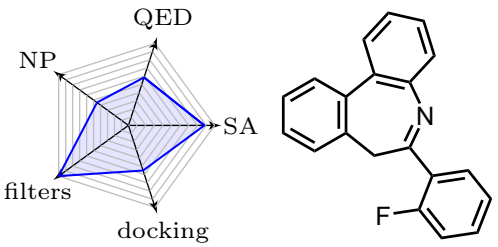


(a) single objective (b) NSGA-II

Figure 4.5: Comparison of the last generation of an exemplary single-objective (ten molecules) and NGSA-II (20 molecules) run. Each line represents a molecule candidate [Cof+20].

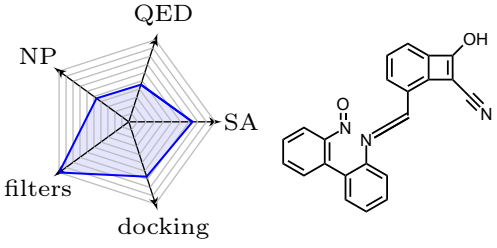
4.5.2 Candidate Comparison

In the following, we will take a closer look at some interesting examples of the evolved molecules. Figure 4.6 presents six molecules with their respective metrics, structural formula, and chemical names. PI-I (a) to PI-III (c) are evolved with the single-objective design, whereas PI-IV (d) to PI-VI (f) show candidates generated by NSGA-II. Overall, we made three major observations during our experiments: Firstly, the algorithm favors molecules with a high amount of aromatic rings. Secondly, the generated molecules tend to be relatively short, especially those with a high drug-likeness. A possible explanation might lay in the functioning of the SELFIES syntax. The token sequence is processed from left to right, and if the syntactic rules reach a terminal state, all the following tokens are ignored. This hampers the formation of longer molecules. Finally, the candidates with the highest binding affinities often had unrealistic geometries. When considering the exemplary molecules in Figure 4.6, all pictured molecules pass the MCF. PI-I offers a reasonable docking score while also scoring high on SA. The molecule PI-II offers an excellent docking score of $-9.7\,\mathrm{kcal\,mol^{-1}}$. PI-III, PI-IV, and PI-VI all score high in terms of the drug-likeness QED and provide good docking results



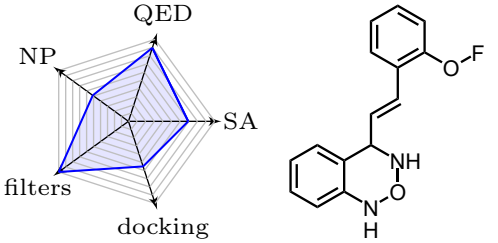
6-(2-fluorophenyl)-7H-dibenzo[b,d]azepine

(a) PI-I, single objective



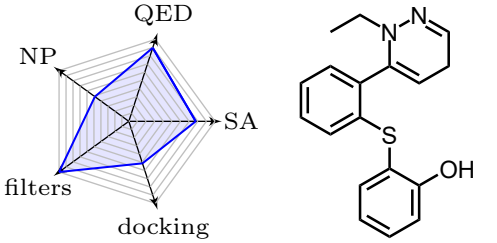
8-hydroxy-5-(2-((2'-nitroso-[1,1'-biphenyl]-2-yl)imino)vinyl)bicyclo[4.2.0]octa-1,3,5,7-tetraene-7-carbonitrile

(b) PI-II, single objective



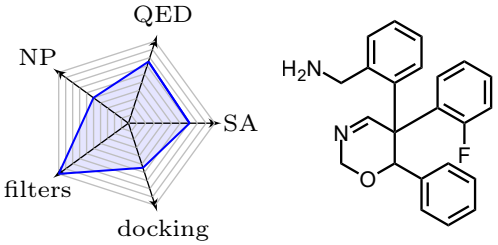
2-(2-(3,4-dihydro-1H-benzo[c][1,2,6]oxadiazin-4-yl)vinyl)phenyl hypofluorite

(c) PI-III, single objective



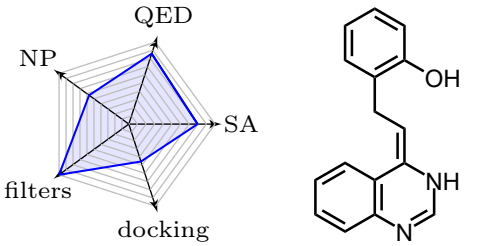
2-((2-(2-ethyl-2,5-dihydropyridazin-3-yl)phenyl)thio)phenol

(d) PI-IV, NSGA-II



(2-(5-(2-fluorophenyl)-6-phenyl-5,6-dihydro-2H-1,3-oxazin-5-yl)phenyl)methanamine

(e) PI-V, NSGA-II



2-(2-(quinazolin-4(3H)-ylidene)ethyl)phenol

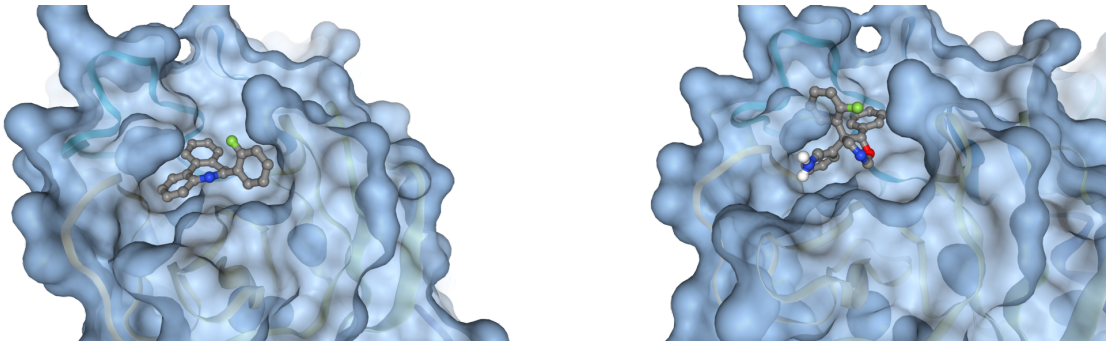
(f) PI-VI, NSGA-II

Figure 4.6: Exemplary protease inhibitors with properties presented as radar plot, structural formula, and chemical name, a-c: single-objective, d-f: NSGA-II results [Cof+20].

around $-7.0\,\mathrm{kcal\,mol^{-1}}$. PI-V offers a good balance between all the five molecule metrics and has a docking score $-7.7\,\mathrm{kcal\,mol^{-1}}$. Figure 4.7 shows a visualization of how ligand PI-I and PI-V are localized in the target pocket of M^{pro}. The position and ligand conformation were optimized with QuickVina 2.

4.6 Conclusion

In this chapter, we addressed the development of a protease inhibitor candidate for the M^{pro} of SARS-CoV-2 as a multi-objective optimization task considering five different molecular metrics. Therefore, we employed a GA that utilizes SELFIES for molecule representation and QuickVina 2 for binding affinity prediction. The algorithm either combines the five metrics



(a) PI-I in M^{pro} pocket (b) PI-V in M^{pro} pocket

Figure 4.7: Molecules PI-I and PI-V docked to the pocket of SARS-CoV-2's M^{pro} [Cof+20].

to a single weighted sum and optimizes them jointly or leverages NSGA-II to approximate a set of molecules with a broad distribution of metrics on the Pareto front. We demonstrate that the algorithm is able to evolve promising molecules, which could form a starting point for further analysis. Although the metrics presented are based on heuristics, and the resulting molecules are not guaranteed to be suitable inhibitor candidates, the presented approach could facilitate the early stages of the drug discovery process as an AI-assisted virtual screening of the chemical biomolecule space.

All in all, we conclude that the chosen SELFIES representation and the chosen mutation operations are suitable for multi-objective molecular design. However, further work could incorporate strategies to prevent the generation of bloated SELFIES, i.e., long strings that describe comparatively small molecules. In addition, integration of more advanced multi-objective evolutionary algorithms like the SMS-EMOA [BNE07], and improved tools for the binding affinity prediction like AutoDock GPU [San+21] are conceivable.

5 Spatial Generation of Molecules with Transformers

The search for new molecules has a critical part to play in various fields of research and industry, for example, in the domain of drug design or energy production [Pol+20]. However, the space of potential candidate molecules is large, which has led to an increased demand for efficient machine learning techniques for the molecular search problem. Generative models play an essential role in this process, as they allow the creation of new molecules and, thus, the traversal of the search space. The previous chapter already demonstrated how GAs can be applied to molecule optimization. However, even though metrics favoring druglike molecules were integrated into the approach, some generated molecules show unrealistic structures and seem difficult to synthesize. GAs, as iterative stochastic search methods, aim to generate optimal molecules with respect to the optimization goals. Thus, they have no intrinsic incentive to generate realistic molecules.

Generative models based on neural networks have become increasingly popular in recent years for the de novo design of molecules (see Section 3.3). Using large amounts of molecular data, these models learn the chemical principles underlying these molecules and can be used to find new realistic structures. A majority of the models operate on molecules in a string or graph representation to encode a molecule's atoms and bonds. These representations provide a straightforward way of dealing with molecules; however, they are not without drawbacks. There are fewer examples of generative models working directly on atom coordinates. However, these models offer the advantage of incorporating all geometric information. This paper introduces a new generative model for molecules working on three-dimensional point clouds of atoms. The generation process is structured as a sequence-to-sequence task, and a transformer-based architecture is implemented. This enables the model to use the powerful self-attention mechanism to attend to previously generated atoms. In general, the model exhibits the following advantageous features:

- The model operates in the three-dimensional atom space, allowing the model to process and generate molecular structures while taking the geometry into consideration.
- The model is able to sample new molecules that are similar in structural properties to the training data.
- The model is able to complete an arbitrary set of atoms into a valid molecule, which offers new interesting opportunities for application as a generative model.

- The transformer architecture allows for the parallel processing of a molecule's atoms during training, resulting in a fast and scalable training procedure.

The following chapter is based on the published study: "Spatial Generation of Molecules with Transformers" [CTK21].

Outline. In the following, Section 5.1 explains the functioning of transformers and their typical applications. Section 5.2 provides an introductory overview of different types of molecule-generating models. Afterward, in Section 5.3, the introduced transformer-based generation model is described, both in its architecture and in the training and sampling procedure. In Section 5.4, the model is evaluated in its performance of generating new molecules and the capability to complete unfinished sets of atoms to valid molecules. Conclusions are drawn in Section 5.5.

5.1 Transformers

Transformers are a neural network architecture originally introduced for sequence processing tasks. There are several areas where machine learning is applied to sequential data, such as processing sensory and financial data or, as it is typical for transformers, language modeling, and translation. Vaswani et al. [Vas+17] introduced transformers as an alternative to the state-of-the-art techniques for sequence modeling, like recurrent neural networks, long short-term memory, and gated recurrent neural networks. Those strategies rely on recurrence, i.e., the models keep track of a hidden state h. The output for a position t depends on the input element at x_t and the previous hidden state h_{t-1}. This allows the model to condition its output on previous elements of a sequence. However, this design has drawbacks. Firstly, the sequential nature impedes parallel processing, which is especially important for longer sequences. Second, this strategy makes it difficult to capture dependencies between elements located at distant positions in the sequence. The transformer architecture circumvents these disadvantages by relying solely on an attention mechanism to draw dependencies between input and output elements. The following is an overview of the underlying architecture, how the attention mechanism works, and typical applications of transformers.

5.1.1 Transformer Architecture

In its original form, a transformer is based on an encoder-decoder architecture, as shown in Figure 5.1. The encoder processes an input sequence of symbols $(x_1, \ldots, x_n)$ to a continuous encoded representation $z = (z_1, \ldots, z_n)$. The decoder combines z with previously generated symbols to predict output probabilities for the following symbol in the sequence. Before the transformer can process the input or output sequences, they must be converted into a sequence of vectors. This is achieved by applying a learnable embedding to these sequences before passing them to the transformer. The embedding layer maintains a fixed-size continuous representation for each possible input token. These representations can either be trained

directly in conjunction with the transformer or taken from an already well-performing model from a similar task. Depending on the used embedding, the resulting sequence consists of vectors with a dimensionality of d_{model}. Increasing the size of the vectors allows the model to generate more complex representations and can increase the model's predictive power, but it also increases the computational resources needed to train and execute the model.

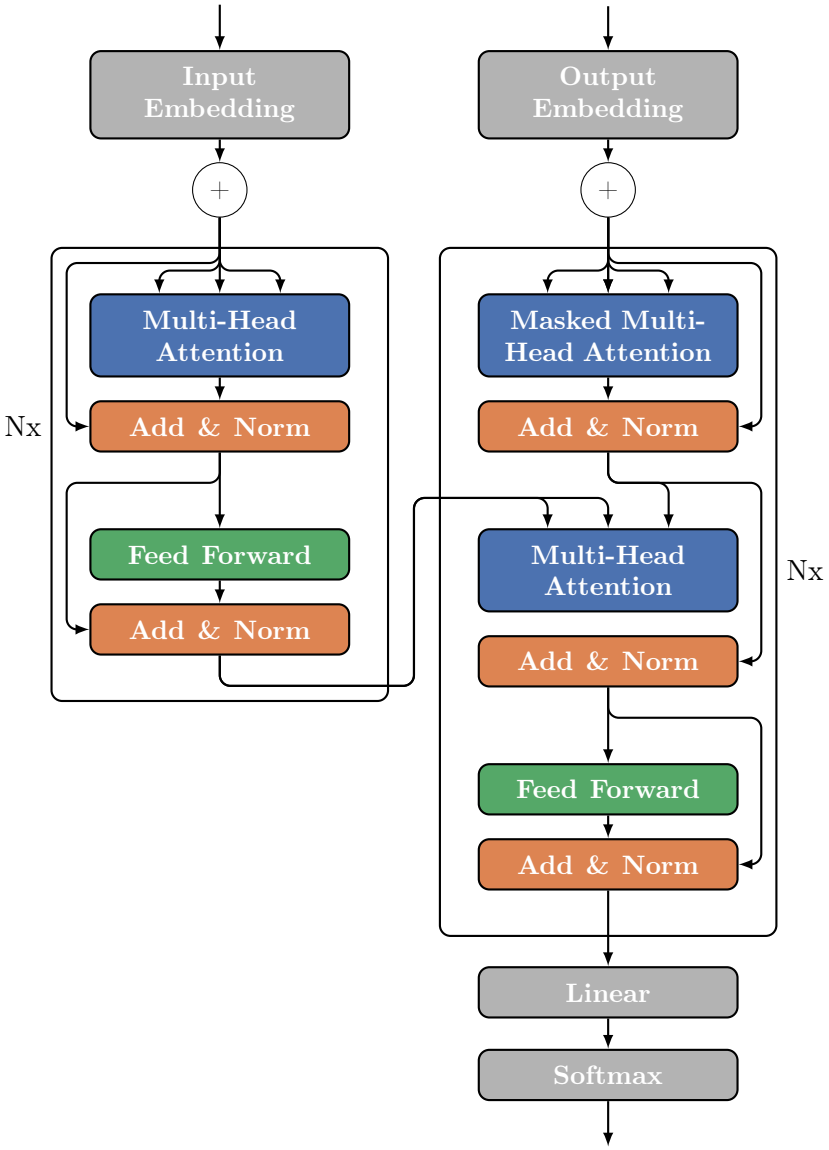


Figure 5.1: Overview of the transformer architecture based on the illustration provided by Vaswani et al. [Vas+17].

Both the encoder and decoder consist of layers that process their input with a combination of the attention mechanism and a feedforward layer. Since each layer outputs vectors of the same dimensionality as its inputs (d_{model}), the architecture allows stacking multiple of

these layers to increase the computational capacity of the transformer. The encoder layer consists of two sequential subcomponents, which feature skip connections and are followed by layer normalization. The first component implements the multi-head attention mechanism. It is followed by a simple, fully connected feedforward network applied to every position of the sequence in parallel. These components are also found in the decoder layer, as well as a masked multi-head attention block for processing the previously generated outputs. By applying a binary mask to the attention mechanism, it is possible to control which elements in the output sequence the transformer can attend to. Considering the example of language translation, the input sentence corresponds to the words of one language and the output to the respective ground truth translation. When predicting the word at position t in the output sequence, the transformer shall only be able to attend to all previously generated words at positions less than t. The decoder features a second multi-head attention block that combines the processed input and output sequence and a feedforward network that generates the decoder layer output. As with the encoder layers, stacking multiple decoder layers increases the processing power of the transformer. The final output of the stacked encoder and decoder layers can be processed by a linear layer and a Softmax activation function to predict the probability for the next token.

5.1.2 Attention Mechanism

Attention was already a concept in language processing tasks before transformers were introduced, e.g., by Bahdanau, Cho, and Bengio [BCB15], but transformers were one of the first models to rely solely on attention rather than recurrence. Intuitively speaking, attention combines the vectors of the input sequence to generate an output sequence of the same length and dimensionality. The attention weights determine the influence of each input vector on the output vectors. At the first step of the attention mechanism, three linear layers process the inputs and generate a query, key, and value sequence. The value sequence of vectors of dimensionality d_v contains the information that will form the output sequence. The queries and keys of dimensionality d_k are combined into the attention weights to determine how much influence the values have on the output sequence. The weights are calculated by the dot-product between the matrix of queries Q and the matrix of keys K and scaled by the factor $\frac{1}{\sqrt{d_k}}$. Scaling is done to accommodate possible large values from the dot product of vectors with a high d_k. At this step, a mask can be applied to the attention weights before each row is rescaled to a sum of one by applying the Softmax function. Multiplying the value vectors V results in the output of the scaled dot-product attention,

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}V\right) \tag{5.1}$$

as described by Vaswani et al. [Vas+17].

The scaled dot-product attention mechanism would suffice for a simple version of the transformer architecture. However, transformers incorporate a more powerful multi-head version of this form of attention. The queries, keys, and values are transformed to a smaller di-

mensionality h times by different linear projections, where h is the number of heads. The smaller dimensionality d_k and d_v are commonly set to $d_k = d_v = d_{\text{model}}/h$ and each head features its own set of learnable linear projections $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$. An independent attention mechanism processes each projected set of queries, keys, and values, and the results are concatenated and combined by a learnable output projection $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$:

$$\begin{aligned} &\text{MultiHeadAttention}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_i)W^O, \\ &\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \end{aligned} \tag{5.2}$$

All heads can perform attention simultaneously, while the reduced dimensionality results in a similar computational cost compared to one scaled dot-product attention on vectors of size d_{model}. However, executing multiple attention operations in one step enables each head to independently attend to different positions in the input sequence, allowing each head to specialize in specific features of the input data.

Attention enables the transformer to process sequential data without the need for recurrence or convolutions. Depending on the generated queries and keys, transformers can adaptively combine vectors from arbitrary positions in the input sequence. Queries and keys can either be generated from the same sequence that provides the value vectors (self-attention) or originate from different sources, like in the decoder's second attention block. Without any additional information, however, the attention mechanism would have no means of taking positional information into account since it is completely based on the content of the vectors. To counteract this, Vaswani et al. [Vas+17] include a positional encoding added to the vectors after the embedding step. The authors decided on a positional encoding based on sine and cosine functions, as in

$$\begin{aligned} \text{PE}_{\text{pos},2i} &= \sin\left(\frac{\text{pos}}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \\ \text{PE}_{\text{pos},2i+1} &= \cos\left(\frac{\text{pos}}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \end{aligned} \tag{5.3}$$

with *pos* being the position of the vector in the sequence and i the dimension. This positional encoding allows the transformer to attend to vectors based on their position in the input sequence and to compare different vectors based on their relative positioning. The authors hypothesize that the latter especially benefits from the chosen sinusoidal functions.

An architecture based on the aforementioned attention mechanism provides advantages over networks featuring recurrent layers or convolutions. First, transformers provide a high degree of parallelizable computation steps compared to recurrent neural networks that process inputs sequentially. Second, transformers simplify the identification of long-range dependencies since the association of any two vectors from a sequence requires only one operation, regardless of their position. In contrast, the number of operations required in a recurrent system is $\mathcal{O}(n)$ and $\mathcal{O}(\log_k(n))$ for convolutional systems, where n is the sequence length and k the kernel size. Lastly, attention, to a certain degree, increases the interpretability of the model. By

analyzing the attention weights, conclusions can be drawn on what inputs are associated and what inputs feature the highest priority for the given task, e.g., [RCW15; Roc+16; Vig19].

5.1.3 Transformer Applications

The advantages of transformers have led to a wide variety of algorithms based on this architecture in recent years. Especially the high degree of parallelizability allowed transformers to be trained on increasingly large datasets, making transformers state-of-the-art in many problem domains. With GPT, a prominent family of transformer-based models for language modeling tasks was introduced by the company OpenAI and subsequently extended with GPT-2 [Rad+19] and GPT-3 [Bro+20b]. These models are trained on large text corpora with an autoregressive objective, e.g., for a sequence of tokens, predict the probability for the next token given all the previous tokens. GPT-2 introduced the idea of pre-training a language model in an unsupervised manner on a large, general text database automatically generated from web pages. The generated WebText database contained approximately eight million documents, resulting in up to 40 GB of text. Although the model was not trained for a specific task, the vast amount of training data and the model's size allowed applying it in a zero-shot fashion to various problems from the language processing domain. Building upon GPT-2, GPT-3 utilizes the same network architecture but with a significantly increased number of parameters (175 billion). Furthermore, the authors increased the dataset size and diversity. The model can be applied to various problems without the need for fine-tuning by presenting the model with a textual representation of the problem.

Devlin, Chang, Lee, and Toutanova [Dev+19] introduced a model called bidirectional encoder representations from transformers (BERT) that founded a different family of language models based on transformers. Unlike the previously mentioned models trained with an autoregressive target, these models can attend to context on the left and right sides of each token. The training procedure contains an unsupervised pre-training phase with masked token prediction, next sentence prediction, and a task-specific fine-tuning phase.

Both approaches have spawned a variety of further research and improved architectures. A comprehensive overview and collection of open-source implementations is, e.g., given by Wolf et al. [Wol+20]. Furthermore, transformer-based models have found wide application in problem domains other than language processing. AlphaFold is an AI developed by Jumper et al. [Jum+21] to solve the protein folding problem. The second version of this AI is based on transformers and trained with a BERT-like masking technique. Tamashiro et al. [Tam+20] demonstrated how transformers can be applied to image processing tasks by translating an image into a sequence of patches, which can be processed by a transformer. The Beijing Academy of Artificial Intelligence presented WuDoa, a multimodal model currently in its second version, which is one of the world's largest AI with 1.75 trillion parameters. It is supposed to be the next step toward artificial general intelligence and can process natural language and images, create realistic artwork and poetry, and compose music.

In summary, due to their powerful attention mechanism and highly parallelizable architecture, transformers find application in various problem domains, especially when they feature large amounts of training data that can be transferred in a sequential representation.

5.2 Related Work: Molecular Generation Models

In the following study, we introduce transformers to the generation of molecules in a spatial representation. The concept of de novo molecule generation has already been featured in Section 3.3. Overall, many generative models for molecules have been developed in the past. Hereinafter, a short overview of different types of models is presented, grouped by the molecular representation on which these models operate.

String-based Representations. The SMILES is a string-based, broadly used representation for molecules [Wei88]. SMILES describes molecules as a sequence of tokens encoding atom types and structural properties, like rings and branches. This allows the application of language processing techniques for molecule generation. For example, Segler, Kogej, Tyrchan, and Waller [Seg+18] used a recurrent neural network to autoregressively construct new SMILES strings. Another example is the work of Gómez-Bombarelli et al. [Góm+18], who used a variational autoencoder in combination with recurrent neural networks on SMILES strings to explore the chemical space.

Although the SMILES representation allows an easy adaption of language processing techniques for molecular search, it is not without drawbacks. A string-based representation can leave out important information about a molecule, like locality [Pol+20]. Additionally, the generation problem is complicated by the fact that generated SMILES strings are not necessarily valid [Góm+18]. Thus, some capacity of the model has to be used to understand the underlying mechanism of the SMILES representation.

Graph-based Representations. Another common approach for the generation of molecules is working on graph-based representations of molecules. A molecular graph is described by a set of vertices representing atoms and edges representing bonds. Hydrogen atoms are often implicitly assumed for simplicity [SG20]. Li, Vinyals, Dyer, Pascanu, and Battaglia [Li+18] introduced a deep generative model for the creation of graphs which they also applied to molecular graphs. The model generates a graph by deciding on a sequence of structure-building actions, like adding a node and connecting it to previously generated ones. They utilized a network architecture based on graph nets to predict the probabilities of the next graph-building operation while considering the existing graph. However, this node-by-node generation can be sensitive to the order of the nodes [SG20]. Additionally, multiple graphs can map to the same molecule, called the graph isomorphism problem that can add noise to the model [Pol+20].

Three-dimensional Representations. By working directly in the three-dimensional space of atoms, a model can utilize the complete information about a molecule without work-

ing on abstract, heuristic concepts like bonds and ring structures. One advantage of such methods is that it enables differentiating between spatial isomers. Furthermore, additional geometric information—like the structure of a protein's active binding site—can be incorporated [Pol+20].

One possible approach for working directly on a molecule's spatial information is positioning atoms in a three-dimensional grid space. Such a representation has already been successfully applied for predicting binding affinities [WDH15; Gom+17]. However, one fundamental challenge of such a representation is the sparsity in the grid space. Only a small portion of the input space is actually used to encode atom information [Kuz+18]. A smoothing transformation can be used to cope with this problem, as shown by Kuzminykh et al. [Kuz+18].

With G-SchNet, Gebauer, Gastegger, and Schütt [GGS19] introduced a generative model operating directly on three-dimensional point sets. The model is able to construct molecules merely based on atoms and their relative positioning. G-SchNet processes a molecule's atoms sequentially, generating new atoms in an autoregressive manner. The order of atoms is determined by a predefined procedure starting from a molecule's center of mass. Molecules are processed by embedding the atom types with a learned embedding and calculating atom interactions based on continuous-filter convolutions. Thereby, the model can process molecules in a translation and rotation-invariant manner. Two auxiliary tokens are used: One marking the atom's center of mass and one shifting the focus to an already placed atom. Newly placed atoms are generated in the neighborhood of the focus token, which can therefore guide the process. G-SchNet was trained on the QM9 database of small organic molecules. The authors demonstrated that the model can generate novel molecules that resemble the training data in their structural properties.

Furthermore, Hoffmann and Noé [HN19] demonstrated that it is possible to utilize neural networks to generate valid Euclidean distance matrices with a three-dimensional embedding. This procedure enabled them to train a Wasserstein GAN that can produce small molecules in a one-shot manner.

5.3 Transformers for Spatial Molecule Generation

This article introduces a novel molecular generative model based on a transformer network architecture. Like G-SchNet, the model can process molecules in a rotation and translation-invariant manner. This is realized by directly operating on the Euclidean distance matrix defined by the distances between molecular atoms. Consequentially, the introduced model shares the earlier mentioned benefits of operating directly in the three-dimensional atom space. The model has direct access to the molecule geometry and can even distinguish between spatial isomers. However, as an advancement over G-SchNet, it does not need any auxiliary tokens and directly predicts atom types and their corresponding distance vectors. Additionally, it is able to process and sample a molecule's atoms in an arbitrary order. This does not just eliminate the need to generate molecule-specific sampling paths but also opens up a wider field of possible applications.

The model generates new atoms in an autoregressive manner. Each molecule is represented by a sequence of atoms $\mathbf{a} = (a_1, \ldots, a_N)$ of length N and the corresponding distance matrix $\mathbf{D}^{N \times N}$ of atomic distance values $d_{i,j}$. Since the distance matrix is symmetric, only the upper-right triangle of the distance matrix is taken into consideration. The model jointly approximates two functions. Firstly, one describing the likelihood of a sequence of atoms types, factorized in the conditional probabilities

$$p(\mathbf{a}) = \prod_{j=1}^{N} p(a_j \mid a_{<j}, d_{:,<j}) \tag{5.4}$$

where $d_{:,<j}$ refers to all distance values of all columns of the distance matrix occurring before the index j. Secondly, the model approximates the likelihood of entry $d_{i,j}$ in the corresponding distances matrix $\mathbf{D}$, in relation to the previously generated atoms and distances

$$p(\mathbf{D}) = \prod_{j=1}^{N} \prod_{i=1}^{j} p(d_{i,j} \mid a_{\leq j}, d_{:,<j}, d_{<i,j}) \tag{5.5}$$

Therefore, the model utilizes the transformer architecture, more specifically the encoder part, to enable attending to already generated atoms and distances. The data structure, architecture, and the training and sampling procedure are introduced in the following.

5.3.1 Data

The architecture requires a preprocessing step to convert a molecule into a representation that the transformer can process. The first step reduces a molecule to its atoms and the corresponding atom coordinates. The coordinates are used to calculate the Euclidean distance matrix. Using the Euclidean distance matrix, rather than the Cartesian coordinates directly, enables the model to process molecules in a rotation and translation-invariant manner.

The set of atoms and the distance matrix are converted to a sequence of atom tokens and a sequence of distances, as pictured in Figure 5.2. Every molecule starts with a *Beginning of Molecule* (BOM) token and is completed with an *End of Molecule* (EOM) token. The input sequences are used as input for the transformer model. The target sequence specifies the expected value at a given index. The model is trained to predict the target values at a given index of the sequence in an autoregressive manner by using all entries in the source sequence up to and including this index. Therefore, the input sequence is generated by passing through the values of the distance matrix's upper right triangle—column by column and row by row. For every distance value, the atom type of the respective column is stored in the atom sequence. At the start of every column, an input pair with a token of the new atom and a placeholder distance value is inserted. This allows the model to take the new atom type into consideration before predicting the column's first distance value. The target distance sequence is constructed by shifting the input distance sequence left by one. The atom target for all distance values of a column j corresponds to the atom type of the $(j+1)$-th column. The model processes molecules in batches to accelerate the training speed. Since molecules

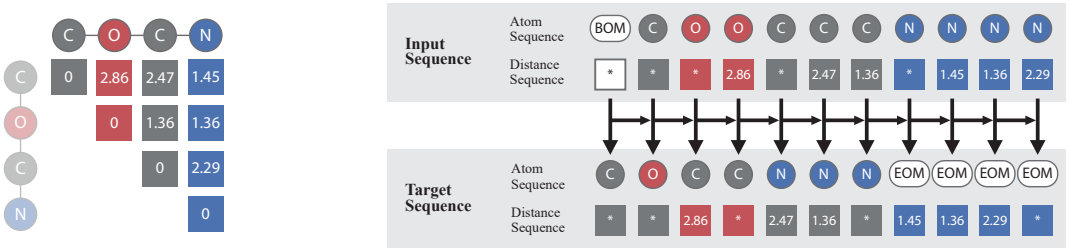


Figure 5.2: Structure of the training data, illustrated by an exemplary molecule. Molecules are defined by a set of atoms and the corresponding distance matrix, pictured on the left. The information is converted to the input and target sequences pictured on the right. Entries of the input sequence marked with an asterisk are placeholders and have no influence on the current prediction. Entries of the target sequence marked with an asterisk are not considered for calculating the loss.

vary in the number of atoms, the input and target sequence are padded to a unified size with PAD tokens, which are processed but ignored for the loss calculation.

For the experiments presented in the following, we used the training and test samples from the QM9 database, as in the work of Gebauer, Gastegger, and Schütt [GGS19]. The database consists of 133 885 stable small organic molecules [Ram+14; Rud+12]. The molecules feature up to nine heavy atoms of the types carbon, nitrogen, oxygen, and fluorine.

5.3.2 Architecture

At its core, the architecture is based on parts of the transformer architecture [Vas+17]. Transformers are sequence transduction models utilizing attention rather than recurrence. Transformers are generally encoder-decoder based. In this work, however, only the encoder part is used. The encoder features a stack of layers implementing a multi-head self-attention mechanism to encode the input sequence to a sequence of continuous latent representations.

The transformer architecture for the generation of molecules builds upon this attention mechanism and is structured as follows. The atom sequence is processed by a trainable embedding layer and concatenated with the distance sequence, resulting in a sequence of feature vectors of size 64. A positional encoding is added to the embedded inputs, which enables the transformer to attend to a desired vector in the sequence based on its position [Vas+17]. The combined atom and distance sequence is processed by six stacked transformer encoder blocks, which utilize eight heads for the multi-head attention to produce the encoded feature vectors. Two decoders, consisting of two fully connected layers and ReLU non-linearity, are used to predict the logits for the respective next atom token. For the prediction of distances, the space between zero and a chosen maximum distance is split into a fixed number of bins. Rather than predicting the distances directly, the distance decoder predicts the index of a distance bin. This procedure yields two benefits: Firstly, the distance values are most likely distributed multimodally [GGS19], which can render regressing continuous targets difficult. Secondly, this enables the sampling of distance values during the generation of

molecules, which can lead to more diverse molecules. A Softmax activation function processes the atom and distance logits to produce the respective probabilities.

5.3.3 Training

The format of the training data renders the training procedure simple. During training, the atom value tuples of the input sequence are processed in parallel. An autoregressive mask prevents the model from attending to values at a later position in the sequence. Furthermore, this procedure implements teacher forcing, meaning that the model always operates on the ground truth data when predicting the probabilities for the next entry. One-hot labels are calculated for the target sequence's atoms and distance bins, and the cross entropy between the predicted probabilities and the target values is calculated. All models mentioned in the following are trained with the Adam optimizer.

For our experiments, we employed a learning rate of 10^{-4} and a batch size of 512 samples. The training was stopped when the loss on the test samples stagnates, which was the case after approximately 100 000 Iterations.

5.3.4 Sampling

The model can sample new molecules or complete existing ones. Since training samples are processed in completely random permutations, the model is able to work with a starting set of atoms of arbitrary size and position. Every molecule is initialized with a BOM token, followed by an optional initial set of atoms and distances. The model sequentially predicts probabilities for the next atom and distance value in an autoregressive manner. As described by Gebauer, Gastegger, and Schütt [GGS19], a temperature parameter can be applied to the Softmax functions to control the amount of randomness. After an atom type and distance value are drawn from the predicted distribution, the input values are appended, and the process is repeated for the extended inputs. The model can generate molecules in a batch to accelerate the generation of multiple molecules and terminates after a configurable maximum number of atoms is generated. For every molecule, only those atoms and distance values are considered, which occur until the first occurrence of an EOM token. The distance values are regrouped into a distance matrix. As it is possible that no set of three-dimensional coordinates satisfies the generated distance matrix, we employ multidimensional scaling to iteratively approximate a suitable set of coordinates. The resulting coordinates and the sampled atom types are passed to RDKit, to allow further processing, like validation of valences and saving the molecules to a common molecular representation.

5.4 Experiments

In the following, we present two experimental conditions to evaluate the generative capabilities of the algorithm. The first condition features a fixed ordering of the atoms in the training molecules. The order is determined by calculating a molecule's center of mass and sorting the

Table 5.1: Results of the generation procedure for the two experimental conditions grouped according to the quality of the molecules. Valid molecules show correct valences and are fully connected. Novel molecules are those not occurring in the QM9 database.

	Fixed atomic order	Random permutations
Correct valence	95.9	93.0
Valid	85.2	59.6
Unique	78.2	58.4
Novel	57.3	48.2

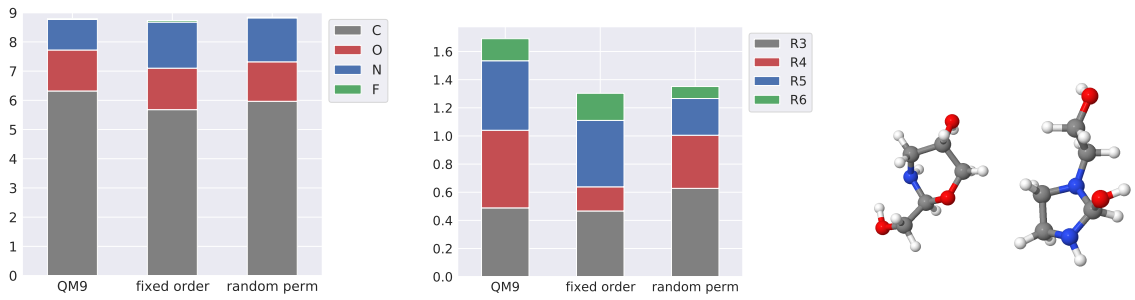
atoms ascending to their distance to the center. This condition forms a baseline to evaluate the model's general capability of generating molecules resembling the training data. In the second condition, a random permutation of a molecule's atoms is drawn each time it is selected as a training sample. Although this should be a more complex task in general, it should train the model to sample atoms in an arbitrary order, allowing the completion of molecules starting with an arbitrary set of atoms.

We trained all models on 50 000 randomly selected molecules of the QM9 database. In the following, the model is evaluated in its capability of generating new molecules from scratch and completing a predefined, unconnected set of atoms placed in a three-dimensional space.

5.4.1 Generating Molecules from Scratch

To evaluate the performance of the models in generating new molecules we sampled 20 000 molecules per condition. The valences of the molecules were validated using RDKit, and molecules with disconnected parts were excluded. Molecules passing these criteria were considered *valid*. Furthermore, all duplicates were excluded from the generated molecules, and those remaining were considered *unique*. Finally, all generated unique molecules were compared to those present in the QM9 database, and only completely new structures remained and are referred to as *novel*. Duplicates were identified by comparing the InChIstring representations of the molecules, which were obtained with RDKit. Table 5.1 presents a comparison of the aforementioned metrics for the two experimental conditions. Most molecules generated by both models pass the valency checks. The model trained on a fixed ordering generated mostly fully connected molecules, resulting in a high amount of valid and unique molecules. Overall, it was able to generate 57.3% completely new molecules, i.e., molecules not included in the QM9 dataset. The model trained on random permutations of molecules exhibited a considerably higher amount of disconnected molecules. It is evident that this effect is a drawback of the training procedure, which features a molecule's atoms in an arbitrary order and is, in general, a more difficult task. Although the number of valid molecules is lower in this condition, nearly all generated valid molecules are unique, with a high portion of those considered novel. Since this model is not dependent on a specific atom ordering, it seems capable of generating more diverse molecules.

In the following, we analyze the generative models' capability of capturing the structural statistics of the training data. Therefore, we followed the procedure described by Gebauer,



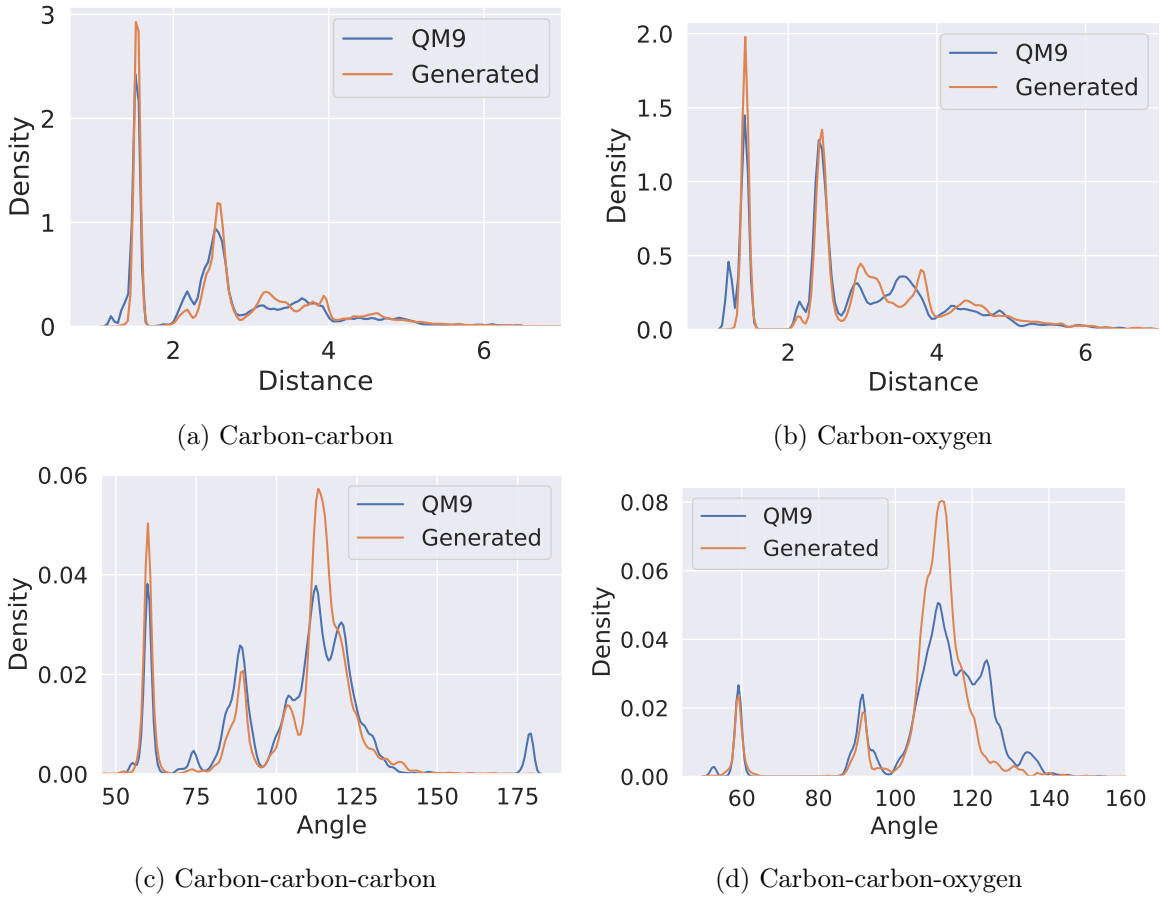
(a) Atom numbers (b) Ring sizes (c) Examples

Figure 5.3: Comparison between the QM9 database and the two experimental conditions on the average number of atoms and rings per molecule. a) The mean number of atoms per molecule grouped by the four heavy atom types: carbon, oxygen, nitrogen, and fluorine. b) The mean number of rings per molecule grouped by the ring size. c) Two exemplary generated molecules.

Gastegger, and Schütt [GGS19] and calculated the mean atom and ring counts of the valid molecules from the two experimental conditions, which were compared to those of the QM9 database. The distribution of atom counts and ring sizes is pictured in Figure 5.3. Comparing the mean number of atoms, pictured in Figure 5.3a, both models seem to generate molecules with a similar total number of atoms compared to the QM9 database. Overall, the distribution of atom types matches, except for nitrogen atoms occurring slightly more frequently in molecules of the generative models. The mean number of rings is pictured in Figure 5.3b, and the comparison reveals that all types of ring structures occurring in the database are also found in the generated molecules. However, the distributions in which the rings of different sizes occur seem to differ between the experimental conditions. The model trained on fixed atom orders generated molecules with a similar amount of rings of the size three, five, and six as in the QM9 database. However, rings of size four occur less frequently in molecules generated in this condition. Molecules generated by the random permutation model seem to feature slightly fewer rings of size four, five, and six and slightly more rings of size three on average.

To analyze how well the distribution of distances predicted by the generative models matches that of the training data, a randomly selected subset of 1000 molecules from the random permutation condition was analyzed consistent with the QM9 database using the Gaussian version of the hybrid DFT functional B3LYP [Bec93; LYP88] and a 6-31G(2df,p) basis set [DHP71; FPB84; HDP72; Kri+80] as implemented in the quantum chemical software package ORCA 4.2.0 [Nee12; Nee18]. We applied a full geometry relaxation using the default convergence thresholds within ORCA.

The relaxed molecules were compared to an equal-sized, randomly chosen subset of the QM9 database, and the results are pictured in Figure 5.4. Atoms of the relaxed molecules should feature distance and angle distributions similar to the training data. Therefore, similar to Gebauer, Gastegger, and Schütt [GGS19], the radial distribution functions of the distances between the two most common atom pairs, namely carbon-carbon (Fig. 5.4a) and carbon-



(a) Carbon-carbon

(b) Carbon-oxygen

(c) Carbon-carbon-carbon

(d) Carbon-carbon-oxygen

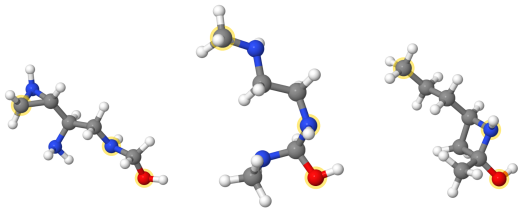
Figure 5.4: Comparison of radial and angular distribution functions between QM9 and the generated molecules. a) Distances between carbon-carbon atom pairs. b) Distances between carbon-oxygen atom pairs. c) Angles between bonded carbon-carbon-carbon chains. d) Angles between bonded carbon-carbon-oxygen chains.

oxygen (Fig. 5.4b) are compared. Furthermore, Figure 5.4c and Figure 5.4d show the angular distribution functions for bonded carbon-carbon-carbon and carbon-carbon-oxygen chains. Comparing the distribution functions, it is apparent that the structures of the generated molecules clearly show similar properties to those of the training data. There seems to be no distinct difference regarding the investigated atom distances when considering the radial distribution functions. The angular distribution functions also show a similar distribution of angles in the database and the generated molecules, with aligning peaks. However, angles between 100 and 120 degrees are slightly overrepresented in the generated molecules.

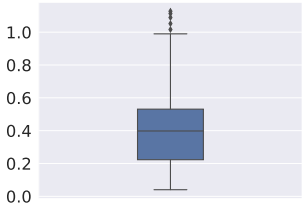
In conclusion, both models are capable of generating new valid molecules simply by placing atoms in a three-dimensional space. The model trained on a fixed atomic ordering produced more valid and slightly more novel molecules, which may be caused by the lower complexity of the task. However, processing atoms in an arbitrary order should give the model more flexibility when it comes to the generation of molecules, which is investigated in the following section. The generated molecules match the training database in many aspects, indicating that the models can identify and reconstruct the underlying pattern of structural properties.

C	O	N
0	6.97	5.07
	0	2.30
		0

(a) Predefined atom distances



(b) Examples



(c) Distance error

Figure 5.5: Results of the completion experiment. a) The predefined distances to be completed (measured in angstrom). b) Some exemplary generated molecules after the relaxation procedure. The predefined molecules are marked in yellow. c) The mean absolute error in angstrom between the three predefined distance values and the real distances after relaxation.

5.4.2 Completing Molecules

Due to processing random atom permutations during training, the generative model should be able to complete an arbitrary set of atoms. Ideally, the resulting molecules should feature similar distance values to those specified, even after relaxation. To investigate this, the model was used to generate 100 valid, novel molecules starting with a set of three predefined, unconnected atoms. The experimental results are pictured in Figure 5.5. Figure 5.5a shows the given distances and the corresponding atom types. These atoms correspond to a randomly selected subset of an exemplary molecule chosen from the QM9 test set. After sampling the atoms and distances, a weighted version of multidimensional scaling [CPH06] is used to obtain the atom coordinates, which allows assigning higher weights to the predefined distance values. Even though the predefined atoms are disconnected, the model trained on random permutation is able to generate approximately 34% valid molecules. As expected, the model trained on fixed permutations cannot generate valid molecules for this configuration. After generation, 100 unique molecules are selected and analyzed with the relaxation procedure described in Section 5.4.1. Figure 5.5b pictures three exemplary molecules after relaxation, with the predefined atoms marked in yellow. To measure how accurately the given distances match the ground-truth data obtained from the relaxation procedure, we compute the mean absolute error, pictured in Figure 5.5c. The median of 0.4 Å indicates that the model is capable of generating multiple different molecules that approximate the predefined distances, even after relaxation.

5.5 Conclusion

In this work, we introduce an autoregressive generative model based on the transformer architecture that can sample new molecules directly in the three-dimensional atom space. The generated molecules show similar structural properties when compared to the original molecules from the training dataset. Training on random permutations of a molecule's atoms

eliminates the necessity of predefined orderings and allows for a broader range of possible applications. For example, extending the model for sampling longer molecules and applying it to the domain of drug design would be a promising subsequent research topic since, in contrast to other approaches, the presented model could take the structure of a protein's binding site into consideration. Overall, a model capable of completing unfinished molecules could accelerate the search for suitable molecular candidates in applications where parts of the target structure are known.

To improve the model, it is conceivable that generating atom and distance values block-wise, rather than value after value, could drastically increase the sampling complexity. A similar approach was shown by Liao et al. [Lia+19] for the generation of graphs with graph recurrent attention networks. Furthermore, Hoffmann and Noé [HN19] demonstrated that it is possible to predict valid Euclidean distances with a neural network directly. An adaption of their approach for autoregressive molecule generation eliminates the need for multidimensional scaling to obtain the atom coordinates.

6 Transformers for Molecular Graph Generation

In the previous chapter, we presented an adaptation of the powerful transformer neural network architecture for the spatial design of molecules. Since this architecture was initially designed for sequence processing, we transferred the molecules' atoms and their respective Euclidean distance matrix to a sequential format and utilized the transformer to autoregressively sample new molecules. One of the advantages of the transformer architecture is the high degree of parallelizability during training. However, in the recent study, we mainly focused on generating small molecules. The main reason is that the simple flattening of the distance matrix causes the sequence length to grow quadratically as the number of atoms in the molecules increases. Since an autoregressive strategy prevents parallelization during sampling, this impedes the generation of longer molecules. In the study presented in this chapter, we explore how we can extend the concepts presented previously so that the generation of more complex molecules is possible. In this regard, we will make the following main contributions: Firstly, we demonstrate how the transformer architecture can be utilized to autoregressively sample new molecules in a graph representation. Secondly, we introduce a more efficient sampling strategy that can sample a whole column of a molecule's adjacency matrix at once to allow the generation of larger molecules.

Generative neural networks for graphs have been extensively studied in recent years, e.g., [Li+18; Lia+19; You+18]. Graphs are a common representation used in various problem domains, and generative models can play an important part in discovering new graphs. Furthermore, they can be used to traverse the search space of candidate graphs in the direction of desired properties. Particularly in molecular design, graphs are a common data structure in addition to the string-based representation. Although some graph-based molecule generation models have been introduced in the past, most of these models rely on recurrence to capture the underlying distributions over nodes and edges. In this work, we introduce an autoregressive molecule generation model based on transformers utilizing their self-attention mechanism.

To investigate whether the generative model captures the underlying structural properties of its training data, we evaluate the quality of the generated molecules with MOSES [Pol+20] and compare it to other state-of-the-art models for molecule generation. The results presented in the following are based on the published article: "Transformers for Molecular Graph Generation" [CK21].

Outline. Section 6.1 provides an overview of graph neural networks in general, how they can be utilized for molecule generation, and the application of transformers for molecular graphs. In Section 6.2, we demonstrate how transformers can be modified to train on molecules represented as sequences of atoms and their adjacency matrix. This enables transformers to be used as a molecular generation model. The generated molecules are compared to those generated by other state-of-the-art models in Section 6.3. Section 6.4 concludes the work and provides an overview of possible future work.

6.1 Related Work: Graph Generative Models

Li, Vinyals, Dyer, Pascanu, and Battaglia [Li+18] introduced a graph generative model, which they also adapted to the domain of molecules. They employed an architecture combining message passing networks and recurrent neural networks to autoregressively generate graphs. Building on this, Liao et al. [Lia+19] improved the autoregressive sampling strategy. By sampling a block of one or several rows of the adjacency matrix at a time, the number of necessary decision steps during graph generation is drastically reduced. Since multiple edges are generated simultaneously, a mixture of multiple Bernoulli distributions is used to decide on the occurrence of edges. These mixture components can capture correlations between edges in one block. GraphRNN [You+18] is another approach for the generation of graphs, which uses a graph-level recurrent neural network (RNN) and edge-level RNN for the generation of graphs.

Molecular generation models often operate on string-based molecular representations (commonly SMILES) since these are easy to access and process. Different approaches have been introduced for the generation of molecules as strings, e.g., recurrent neural networks [Seg+18] and variational autoencoder [Góm+18]. Despite its straightforward implementation, a string-based representation bears some disadvantages. Generated SMILES strings are not necessarily valid, and additional model capacity has to be attributed to learning the language's formal rules [Pol+20]. To overcome these limitations, Podda, Bacciu, and Micheli [PBM20] introduced a fragment-based approach for generating SMILES strings. Rather than sampling symbol by symbol, this approach first extracts a vocabulary of fragments from the training data, on which the language model is then trained.

Directly processing and sampling molecular graphs was, e.g., demonstrated by Li, Vinyals, Dyer, Pascanu, and Battaglia [Li+18]. Furthermore, Jin, Barzilay, and Jaakkola [JBJ18] presented a generative model for molecules based on a junction tree variational autoencoder. Rather than generating molecules on an atom-per-atom basis, this approach combines molecular substructures extracted from the training data, ensureing the generation of valid molecules. A few examples of applying the transformer architecture to molecular graphs can be found in the literature. Cai and Lam [CL20] demonstrated that it is possible to use transformers for graph-to-sequence learning. Additionally, the transformer architecture has been utilized for molecular property prediction [CBJ19; Ron+20]. Yoo et al. [Yoo+20] presented a transformer specialized in processing graphs. Nodes are processed as tokens, while the edges are incor-

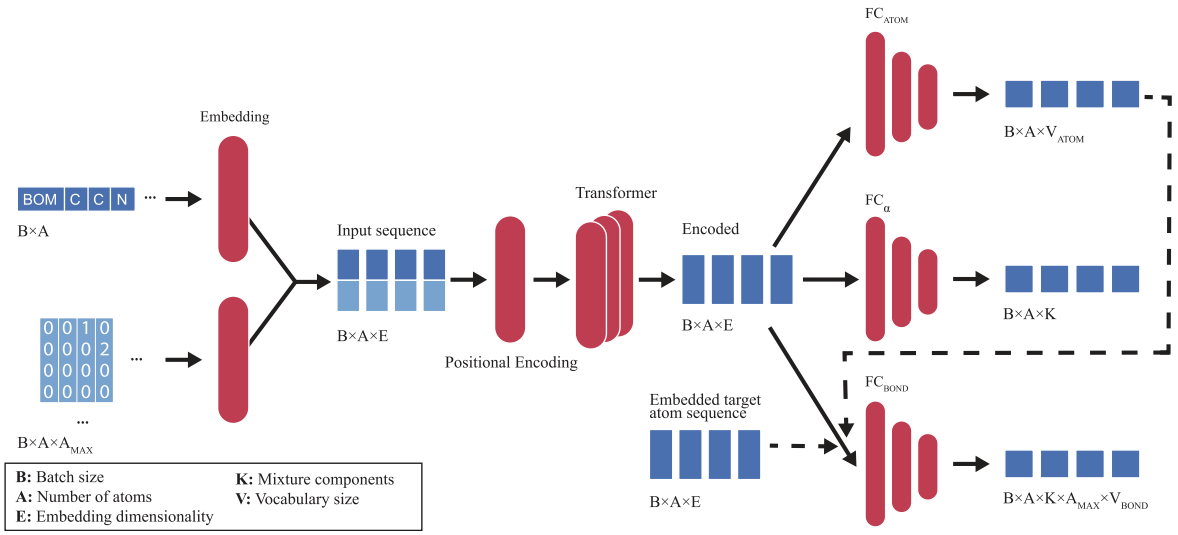


Figure 6.1: Overview of the model's architecture.

porated into the transformer's self-attention mechanism. The model was mainly investigated for property prediction on smaller molecules but can also be applied to graph generation.

6.2 Transformers for Graph Generation

In the following section, we introduce a transformer-based generative model for graphs. Albeit applicable to various types of graphs, in this work, we evaluate its capabilities as a generative model for molecular graphs.

6.2.1 Data Representation

This work utilizes the data sets provided by the benchmark framework MOSES, which are based on the ZINC database and contain 1.6 million training and 176 thousand test molecules [Pol+20]. The molecules are converted to a sequence of atom tokens and a corresponding adjacency matrix. The adjacency matrix marks missing bonds as zeros, while bonds are represented by a number unique to the respective bond type—in this work, single, double, and triple bonds are considered. Since the transformer processes molecules in a column-wise manner, the lower left half of the adjacency matrix is masked with zeros. Additionally, the columns are padded with zeros to a user-defined maximum number of atoms to ensure a fixed number of elements per column. Molecules are processed in batches. If molecules in a batch are of different sizes, the input sequences are padded to the longest molecule length in the batch.

6.2.2 Architecture

The architecture is pictured in Figure 6.1 with an exemplary input molecule. In the initial step, a trainable embedding layer processes the atom sequence. The adjacency matrix is treated as a sequence of matrix columns and embedded by a linear layer. The resulting

two sequences are stacked, and a positional encoding is added, which helps the transformer attend to specific positions in the sequence. This sequence forms the input for the transformer. The transformer consists of multiple stacked encoder blocks and implements the multi-head self-attention mechanism. The encoded sequences are used to predict the target atom and bond logits. However, this process differs between training and inference. In both cases, the probabilities for the next atom type are computed by a fully connected decoder network $\mathrm{FC}_{\mathrm{ATOM}}$. The next column of the adjacency matrix is predicted by combining the encoded sequence with information about the next atom. This way, the model can take a sampled atom's type into consideration when predicting its bonds. During training, this is achieved by passing the ground truth sequence of target atoms to the atom embedding layer and stacking it with the encoded sequence. During sampling, the predicted atom probabilities are used to predict an appropriate next atom type, which is then passed to the atom embedding and stacked with the encoded sequence. In both cases, the atom type enriched sequence is passed to a fully connected decoder network $\mathrm{FC}_{\mathrm{BOND}}$ to predict the next column of the adjacency matrix. Like Liao et al. [Lia+19] proposed, the network predicts not only a single probability per edge but rather a mixture of multiple distributions. This enables the model to capture correlations within one adjacency matrix column. Therefore, outputs produced by the bond decoder have an additional dimension of size K, the user-defined number of mixture components. Since each edge represents a bond and the approach presented considers single, double, and triple bonds, the model predicts a mixture of categorical distributions. A single layer decoder FC_{α} produces the K-dimensional vector of probabilities for these components.

6.2.3 Training and Sampling

During training, the input sequence is processed in parallel. An autoregressive attention mask is used to prevent the model from attending to subsequent sequence entries. This also effectively implements teacher forcing, i.e., the ground truth data of all previous entries in the input sequence is used to predict the next atom type and bonds. The bond decoder $\mathrm{FC}_{\mathrm{BOND}}$ is provided with information about the current atom type when predicting its respective bonds. In this way, the model can dynamically adjust the probability outputs since the conditions for the subsequent bonds are likely to depend on the next atom's type. The way the model incorporates information about the next atom varies between training and sampling, but both strategies are designed to maximize the degree of parallelizability. During training, the atom sequence is shifted left to generate the target sequence. The target sequence is processed by the same embedding layer as the input sequence of atoms, combined with the transformer-encoded sequence, and passed to the bond decoder. During sampling, the model starts with predicting the probabilities for the next atom type. The next atom is then sampled from a categorical distribution, initialized with these probabilities, and also embedded. This information is combined with the already encoded transformer output and passed to the bond decoder, eliminating the need to run the transformer a second time.

The loss is defined by the cross entropy between the predicted atom and bond logits and the respective next atom type in the atom sequence and the next adjacency matrix column. For

Table 6.1: Comparison of molecules created by different generative models.

Model type	Valid	Unique@1k	Unique@10k	IntDiv	IntDiv2	Filters	Novelty
CharRNN	0.9700	**1.0000**	0.9994	0.8562	0.8503	0.9943	0.8419
AAE	0.9400	**1.0000**	0.9973	0.8557	0.8499	0.9960	0.7931
VAE	0.9800	**1.0000**	0.9984	0.8558	0.8498	**0.9970**	0.6949
JTN-VAE	**1.0000**	**1.0000**	**0.9996**	0.8551	0.8493	0.9760	0.9143
LatentGAN	0.9000	**1.0000**	0.9968	0.8565	0.8505	0.9735	**0.9498**
Our Model (fixed)	0.9893	**1.0000**	0.9989	**0.8569**	**0.8510**	0.9943	0.6312
Our Model (depth-first)	0.9751	0.9998	0.9994	0.8568	0.8509	0.9896	0.7931

the bond types, the predicted type and alpha logits are compared to the subsequent column of the adjacency matrix, and the negative log-likelihood is calculated as the bond loss. The bond and atom loss are added and form the loss for the parameter optimization. We optimized the model parameters by gradient descent using the Adam optimizer.

Molecules are sampled in an autoregressive manner. The process is initiated by passing a *Beginning of Molecule* token and an empty adjacency matrix column to the model. After every pass, the predicted probabilities are used to sample the next atom type and adjacency matrix column. The input sequence is then appended with the newly sampled values and passed to the model for the next sampling step. This process is repeated until a maximum number of atoms has been sampled. After sampling, the generated atom sequences are trimmed until the first occurrence of an *End of Molecule* token. The atoms and the adjacency matrix are converted to the desired output format with the help of RDKit.

6.3 Experiments

To evaluate the quality of the generative model, we utilize the evaluation tools provided by MOSES. This allows a comparison to other generative models for molecules, like the CharRNN, adversarial autoencoder (AAE), variational autoencoder (VAE), junction tree variational autoencoder (JTN-VAE) and latent generative adversarial network (LatentGAN) model provided by MOSES [Pol+20]. Our model was trained in two experimental conditions, differing in the order of atoms in the training molecules. In the fixed ordering condition, this order was the same as defined by the data set. However, previous work has shown that graph-based generative models can be applied to various node orderings [Li+18; Lia+19]. Some orderings may be more challenging to learn, but utilizing different node orderings per molecule could lead to a more diverse and robust generative model. Therefore, in a second condition, for every molecule drawn as a training sample, a random starting atom was chosen, and the remaining atoms were sorted by traversing the graph in a depth-first manner.

For evaluation, both models were used to sample the recommended amount of 30 000 molecules each, which were passed to MOSES for analysis. This procedure was repeated ten times, and the mean results are pictured in Table 6.1. Both models generated mostly valid molecules. The model trained on fixed atom orderings is only surpassed by the JTN-VAE, which can only generate valid molecules by design. The fraction of unique molecules in a random

Table 6.2: Similarities between generated molecules and the test/scaffold test set.

	FCD (↓)		SNN (↑)		Frag (↑)		Scaf (↑)	
Model type	Test	TestSF	Test	TestSF	Test	TestSF	Test	TestSF
CharRNN	0.0732	**0.5204**	0.6015	0.5649	**0.9998**	0.9983	0.9242	**0.1101**
AAE	0.5555	1.0572	0.6081	0.5677	0.9910	0.9905	0.9022	0.0789
VAE	0.0990	0.5670	0.6257	0.5783	0.9994	**0.9984**	0.9386	0.0588
JTN-VAE	0.3954	0.9382	0.5477	0.5194	0.9965	0.9947	0.8964	0.1009
LatentGAN	0.2968	0.8281	0.5371	0.5132	0.9986	0.9972	0.8867	0.1072
Our Model (fixed)	**0.0639**	0.5495	**0.6355**	**0.5841**	0.9997	0.9979	**0.9409**	0.0564
Our Model (depth-first)	0.0783	0.5319	0.6148	0.5724	0.9998	0.9981	0.9334	0.0914

subset of 1000 and 10 000 molecules is comparable to those of the other models. MOSES gives two internal diversity metrics, estimating the diversity within the generated molecules and therefore indicating how well the model covers the chemical search space. Both of our models slightly surpassed the other models in these metrics. A high fraction of the generated molecules passes chemical filters (e.g., MCF, PAINS). The model trained on a fixed ordering generated a lower number of novel molecules when compared to the other approaches. As expected, the model trained on different depth-first orderings generated a clearly higher fraction of novel molecules while still generating a high amount of valid molecules. However, some other models still show a substantially higher fraction of novel molecules. It is conceivable that the general necessity of a fixed node ordering limits the model's capability of generating more novel molecules. All in all, in the presented experiments, the proposed approach competed with other state-of-the-art molecule generation models and generated a high fraction of valid and diverse molecules.

Furthermore, MOSES features four similarity measures to determine how closely the generated molecules resemble the test sets. The statistics for the distance measures are presented in Table 6.2. The Fréchet ChemNet Distance (FCD) uses ChemNet and compares the different distributions in the activation of its last layer. The Nearest neighbor similarity (SNN) is defined by the mean similarity of all molecules to their nearest neighbor. Fragment similarity (Frag) and Scaffold similarity (Scaf) define cosine similarities between fragments and scaffold frequencies between the sets. Comparing these similarities, both models generalize well and generate molecules similar to those in the test sets. All in all, the similarity scores are close to those of the other models. Furthermore, the fixed ordering model generated molecules with a better FCD, SNN, and Scaf similarity to the test set and a better SNN similarity to the scaffold test set.

6.4 Conclusion

In this work, we introduce a transformer-based generative model for graphs that directly utilizes the multi-head self-attention mechanism to predict distributions over nodes and edges. Representing a molecular graph as a sequence of atoms and adjacency matrix columns allows a straightforward adaptation of the transformer architecture. Rather than relying on standard

message passing, the transformer can utilize self-attention to decide how information is aggregated at every generation step dynamically. In contrast to approaches relying on recurrence, transformers can process every generation step in parallel during training, which facilitates training on large molecular databases. The presented approach generates a mixture of probability distributions to simultaneously generate all elements of an adjacency matrix column whilst still capturing in-column dependencies. Extending previous approaches, we generate an output of mixed categorical distributions to model the occurrence of edges and their respective binding type. In experiments on the generation of molecular graphs, the model was able to generate a high amount of valid molecules. Different distance metrics suggest that the model generalizes well to unseen molecules and is on par with other state-of-the-art molecule generation models. Due to the parallelizability of the Transformer architecture and the fact that no message passing is required, the framework scales well even for larger problems.

Possible directions for future work include further improving the generated number of valid molecules and the molecules' novelty. The former could be achieved by adding a masking procedure to enforce simple atomic valency rules, similar to what was done by Imrie, Bradley, Schaar, and Deane [Imr+20]. Unconnected nodes could be prevented by enforcing at least one connection to previous nodes in the sampling step. Furthermore, the results indicate that training on different node orderings can increase the number of novel molecules. Incorporating additional ordering heuristics and utilizing varying orderings could artificially increase the number of training samples and result in the discovery of even more new molecules.

7 An Evolutionary Fragment-based Approach to Molecular Fingerprint Reconstruction

Artificial intelligence has an increasingly important part to play for in silico drug design. Choosing an appropriate representation of biomolecules is a crucial first step in the modeling process. Various representations are utilized to store and process molecular data. One commonly employed approach is to use string-based representations, like the SMILES. SMILES encodes the graph of a molecule by specifying atoms, bonds, ring structures, branches, and other chemical information as characters in a string. Processing molecules in a string representation allows straightforward adaptation of language modeling techniques for generating new molecules, as, e.g. demonstrated by Segler, Kogej, Tyrchan, and Waller [Seg+18]. Since SMILES strings follow strict syntactic rules, it is possible to generate invalid strings or molecules with incorrect valences. Operating on fragments of SMILES strings, rather than on a character-by-character basis, can facilitate the generation of valid molecular structures, as shown by Podda, Bacciu, and Micheli [PBM20]. A further way of representing molecules is using graph structures of atoms and their connecting bonds. This representation allows the application of graph neural networks for processing and generating molecules. Li, Vinyals, Dyer, Pascanu, and Battaglia [Li+18], for example, demonstrated how, with the help of these networks, molecular graphs can be constructed by iteratively generating atoms and connecting them to previous nodes in the graph. Further examples are LigBuilder (a program for the design of ligands) introduced by Wang, Gao, and Lai [WGL00] and the GA introduced by Jensen [Jen19], which operate directly on molecular graphs.

This work focuses on a different family of representations for chemical structures: molecular fingerprints. Fingerprints encode the presence of molecular features, often in the form of a fixed-size bit vector. The resulting vectors can be utilized to approximate the similarity of molecules and are therefore commonly used to search molecule databases for structures similar to a target molecule. Furthermore, they can be used for substructure searching, clustering, and classification tasks [RH10]. Various fingerprints are available, which differ in the information they encode. Topological fingerprints like the RDKit fingerprint consider all subgraphs in a molecule up to a specific length and hash information about atom types, atomic numbers, aromaticity, and bond types. The more recently developed circular fingerprints consider the environment around every atom up to a specific bond radius [RL13].

Although fingerprints are applied for various tasks, there are fewer examples of using fingerprints for molecule generation and optimization. One potential reason is that fingerprints

are typically not decodable [Pol+20]. There is no direct strategy for finding the molecule from which a particular fingerprint originates. Additionally, multiple molecules can potentially map to the same fingerprint [Le+20].

An early example of using GA for the reconstruction of molecular descriptors has been given by Masek, Shen, Smith, and Pearlman [Mas+08]. They demonstrated on a small number of target molecules and for a simple set of descriptors how a GA can be used to generate molecules with similar descriptors. Winter, Montanari, Noé, and Clevert [Win+19] provided one example of a reconstructable molecule representation. They introduced a deep neural network for the translation of molecular representation into another. Using a small, fixed-size latent vector between the encoder and decoder, they forced the model to encode meaningful information in this vector, which can be utilized as a continuous and data-driven molecular descriptor (cddd). Building up on this, Le, Winter, Noé, and Clevert [Le+20] trained a neural network to predict the cddd from a given circular ECFP, which is a commonly used type of fingerprint. By first translating the ECFP into the cddd-representation and then using the model introduced by Winter, Montanari, Noé, and Clevert [Win+19], they were able to reconstruct a portion of the original molecules from their respective fingerprints. Even if some molecules could not be reconstructed entirely, the resulting molecules showed similar features compared to the original molecule. Kwon, Kang, Choi, and Kim [Kwo+21] trained a recurrent neural network to directly predict a molecule's SMILES representation from its corresponding ECFP. This enabled them to apply a GA for molecule optimization tasks. Implementing individuals as fixed-size bit vectors allowed for a straightforward adaptation of classic genetic operations to this problem. For the fitness evaluation, the molecule's phenotype—in the form of a SMILES string—was deduced by the aforementioned RNN. This application demonstrates the potential of a method capable of generating molecules from its fingerprint representation.

Working with molecules in their fingerprint representation offers numerous advantages. Fingerprints are fast to calculate and allow similarity measurements between molecules. Their fixed size allows for a straightforward application of GAs and various neural network architectures. Reliable methods are needed to decode fingerprints into molecules similar to their origin molecule or even reconstruct them entirely. This study presents such a method in the form of a GA, capable of combining molecule fragments to decode molecular fingerprints. The algorithm bears the following advantages:

- The algorithm constructs molecules from a set of fragments rather than on the scope of individual atoms.
- By utilizing the *breaking of retrosynthetically interesting chemical substructures* (BRICS) algorithm for the creation of fragments, the generated molecules are valid and composed of retrosynthetically interesting substructures.
- We demonstrate the algorithm's capability of reconstructing molecules from their fingerprints. Even if a molecule is not fully reconstructed, the generated molecules show similar structures to the original molecules.
- The algorithm can generate molecular structures of arbitrary size.

This chapter is based on the following published article: "An Evolutionary Fragment-based Approach to Molecular Fingerprint Reconstruction" [CK22].

Outline. This paper is structured as follows: Section 7.1 presents the GA approach for fingerprint reconstruction by introducing the fragment-based representation, the data used to construct the set of fragments, and the genetic operators applied during evolution. The approach is experimentally analyzed in Section 7.2, complemented by a transformer-based approach for fingerprint generation. In Section 7.3, the findings are concluded, and an outlook on possible further work is presented.

7.1 Genetic Algorithms for Molecular Fingerprint Reconstruction

The GA introduced hereafter constructs molecules and optimizes them to match a respective target molecular fingerprint. In the following, we will describe the general structure of the GA, how individuals, i.e., the molecules, are represented in the algorithm and the applied genetic operations.

Algorithm 2 Pseudocode of the Fingerprint GA

```
1:  f_target ← read_target_fingerprint()
2:  P ← initialize(pop_size)
3:  repeat
4:      for all i ∈ P do                                  ▷ fitness evaluation
5:          smiles_i ← i.map_phenotype()
6:          f_i ← calculate_fingerprint(smiles_i)
7:          i.fitness ← T(f_i, f_target)
8:      clearing(P)                                       ▷ apply niching
9:      P_new = select_dominant_individuals(P)
10:     repeat                                            ▷ generate a new child
11:         p_1 ← tournament_selection(P)
12:         p_2 ← tournament_selection(P)
13:         c ← crossover(p_1, p_2)
14:         c.mutate()
15:         P_new.append(c)
16:     until len(P_new) = pop_size
17:     P ← P_new
18: until termination_condition()
```

7.1.1 Genetic Algorithm

The structure of the GA is pictured in Algorithm 2. It starts by generating an initial population of individuals representing candidate molecules and evaluating their fitness. The fitness is evaluated with regard to the respective target fingerprint f_{target}. Firstly, a molecule is mapped to its phenotype in the form of its SMILES representation. Afterward, RDKit is

used to generate the molecule's fingerprint f_i and to measure similarity to the target. The algorithm is generally independent of the chosen fingerprint type and bit vector size. The similarity is calculated by the Tanimoto similarity $T(f_i, f_{\text{target}})$ (also known as the Jaccard index) between the individual's and the target's fingerprint and ranges between zero (dissimilar) and one (similar).

It is conceivable that this fitness function results in a multimodal fitness landscape, e.g., different possible solutions for a respective target fingerprint exist. Therefore, we employ a niching strategy to keep the population diverse and prevent premature stagnation. In principle, many niching strategies could be applied to this problem. Since molecules are converted to their fingerprints for fitness evaluation, the Tanimoto similarity offers an uncomplicated way of measuring the distance between individuals, which is a core component of many niching techniques. We employ a clearing strategy [Pet96] in our featured approach. Clearing divides the population into subspecies. Individuals in one species compete for limited resources, whilst different species can coexist without competition. Each subpopulation is centered around a dominant individual with the best fitness. A configurable similarity threshold defines the minimum distance between dominant individuals. All individuals more similar than this threshold belong to the same species. Clearing assigns a fitness of zero to all individuals in the same species, except for a configurable number of the specie's best individuals.

After the fitness evaluation and niching are completed, the next generation is constructed. We employ elitism, e.g., we transfer the dominant individual of every subspecies unaltered into the next generation. The new population is constructed by repeatedly selecting two parents by tournament selection, from which a new child is created by crossover. The child is mutated by one of the multiple possible mutation operators and inserted into the next generation. The cycle of generations is repeated until the termination criterion is met.

7.1.2 Representation and Molecule Data

As typical for many generative models for molecules, we represent molecules as graphs. A molecular graph is defined by a set of nodes representing atoms of different types and edges representing bonds between atoms. The bonds can also be of different types, e.g., single, double, triple, or aromatic. However, constructing molecules in an atom-by-atom manner yields disadvantages. For example, it allows the generation of molecules with incorrect valences. It may be more efficient for the algorithm to work directly with meaningful substructures rather than having to identify and assemble them. Finally, working with fragments of molecules rather than atoms requires fewer generation steps per molecule and can scale well to the generation of larger molecules.

One example of a fragment-based generative model constructing molecules has been given by Jin, Barzilay, and Jaakkola [JBJ18]. They first created a set of fragments from a molecule database and then trained a JTN-VAE to encode molecules to a tree-structured scaffold of substructures. The nodes are then decoded by a message passing neural network and assembled into a valid molecule. Pegg, Haresco, and Kuntz [PHK01] introduced a GA for the generation of new ligands for a given protein target for which they represented molecules as an

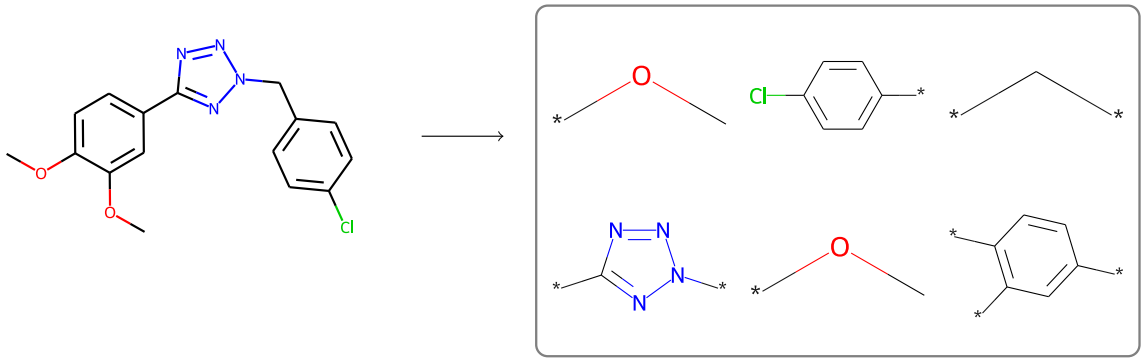


Figure 7.1: The BRICS fragments for an exemplary molecule. Asterisks mark breaking points between fragments.

acyclic graph consisting of substructures from a small set of fragments. Podda, Bacciu, and Micheli [PBM20] presented a language model for molecular design. Rather than operating directly on the SMILES representation and processing sequences of characters, they trained the model on a sequence of molecule fragments. Due to this strategy, the model solely generates valid molecules. As molecules must be reconstructible from a sequence of fragments, they introduced a specific fragmentation strategy. This strategy requires a fixed ordering of a molecule's atoms. Molecules are recursively split into two fragments. The left is kept unchanged, while the right is further split. Although this procedure ensures that molecules are reconstructable, it reduced the average number of fragments per molecule to 2.24. To generate the set of fragments used by their model, they utilized the BRICS algorithm [Deg+08], which is also featured in this study. This algorithm for obtaining fragments from biologically active compounds breaks molecules at retrosynthetically relevant bonds. An example of a molecule and its BRICS fragments is given in Figure 7.1. Asterisks represent dummy atoms that mark possible connection points for each fragment.

Table 7.1: Statistics of data set and resulting fragment library. N_c denotes a fragment's maximal number of connections to other fragments.

Number of molecules	1 526 990
Total number of fragments	31 102
Number of fragments $N_c = 1$	16 948
Number of fragments $N_c > 1$	14 155
Avg. fragments per molecule	4.88
Avg. atoms per molecule	24.53

Individuals of the GA featured in this study are composed of such fragments. The algorithm can work with an arbitrary set of fragments, and fragments can be included or excluded depending on the chosen problem domain. The experiments presented in this study are based on a fragment library derived from the data provided by the MOSES framework for molecular generation models [Pol+20]. This framework provides training and test data sets based on the ZINC molecule library. We composed the fragment library by iterating over the training data

set, consisting of around 1.6 million molecules. RDKit was used to decompose every molecule into its BRICS fragments, resulting in fragments with a varying number of connection points N_c. Molecules consisting of only one or two fragments were excluded. All other fragments were composed in a fragment library, keeping track of the frequency of their occurrence. The descriptive statistic about the data set is given in Table 7.1.

Inspired by genetic programming, individuals are represented by a tree structure. Every node is described by one BRICS fragment. The tree's leaves feature fragments with only one connection point. Non-leave nodes are defined by one fragment with more than one connection point. Each node can have up to $N_c - 1$ child nodes. The last remaining connection point connects the substructure to its parent. The connection points are enumerated to define the order in which they are connected to the parent and child fragments. The tree's root node is a dummy node that always has two children and connects the tree's left and right halves. To assemble a molecule, the fragments are connected as defined by the tree structure. For further processing, the resulting graph can be converted to a SMILES string by RDKit. This strategy ensures that the resulting molecular graphs are coherent and valid with respect to atomic valences. Furthermore, it allows an easy implementation of techniques and genetic operations such as those used in genetic programming, which are described in the following.

7.1.3 Genetic Operators

Due to the similarity between the representation of program trees in genetic programming and our chosen molecule fragment tree representation, we adapted principles and genetic operations from genetic programming to our problem. The initial population of individuals is created by the *grow* method. Starting at a tree's root node, random fragments are chosen from the set of fragments and added to their parent node. If a predefined maximum depth is reached, only terminal nodes (those with only one connection point) are sampled. Crossover is implemented as a subtree crossover between the two parent molecules. The fragment trees are split at their root node, and the child is created by combining one parent's left half with the other parent's right one. Multiple mutation operators are included to allow a finer adjustment of the molecules during evolution:

Insertion selects a random non-leave node and inserts a random fragment with at least two connection points between the chosen node and one of its child nodes.

Deletion selects a random non-leave node and deletes it from the tree. The node's first child is connected to its parent to preserve a portion of its subtree and limit this mutation's effect on the molecule.

Permutation selects a random non-leave node and randomly permutes the ordering of its connection points. This can change the connection point at which the fragment is connected to its parent and the points connected to its children.

Point mutation: Frequency selects a random node and replaces its fragment with one from the set of fragments. The probability for each fragment is weighted by its frequency in the data used for constructing the fragment set.

Point mutation: Similarity also selects a random node and replaces its fragment with a random one. All previous mutation operations can have a significant impact on the resulting molecule. To allow a finer adjustment of already well-performing individuals, the probability of a fragment being chosen as a replacement is weighted by its Tanimoto similarity to the replaced fragment. The similarity is calculated by comparing the fingerprint of the fragment to be mutated with the fingerprints of every fragment in the fragment library.

7.2 Experiments

In the first part of this section, we investigate the GA in its capability of reconstructing molecules for a given fingerprint. In the second part, we present a case study on a novel generative model that can sample new molecular fingerprints. The introduced GA is used to decode the fingerprints to molecular structures, which combines the two approaches to a new molecular generation technique.

The default configuration for the GA, which is derived from preliminary experiments, is described in the following: The population consists of 300 individuals. These are initialized by the *grow* method with a maximum depth of two. Individuals are selected for reproduction by tournament selection, with a tournament size of ten. When a new individual is created, it is mutated with a random mutation operator. Each operator is weighted to control the probability of its selection. We weighted all mutation operators equally, except for the similarity-based point mutation. Since this operator is the algorithm's best tool for making minor adjustments to already well-performing individuals, the operator is weighted four times as high as the others. Niching is configured to have a clearing radius of 0.6 and a capacity of four per niche. If a niche contains more than four individuals, the niching operator clears their fitness scores.

7.2.1 Reconstruction of Molecules

As mentioned, we used the train and test data set provided by MOSES, which are based on the ZINC database of drug-like molecules. Although the GA does not require training—as neural networks do—we maintained this split and only used the training data to create the fragment library. The molecules from the test set were used for the following evaluations.

To investigate if the introduced GA can reliably improve molecules in the direction of a target fingerprint, we selected nine random fingerprints from the test data and carried out 30 independent optimization runs per target molecule. The target molecules were converted to their fingerprint representation with RDKit. We used RDKit's Morgan fingerprint—the RDKit implementation of the common ECFP—with a bit vector size of 4096 and an atom radius of six. During the fitness evaluation, fingerprints for the generated molecules were

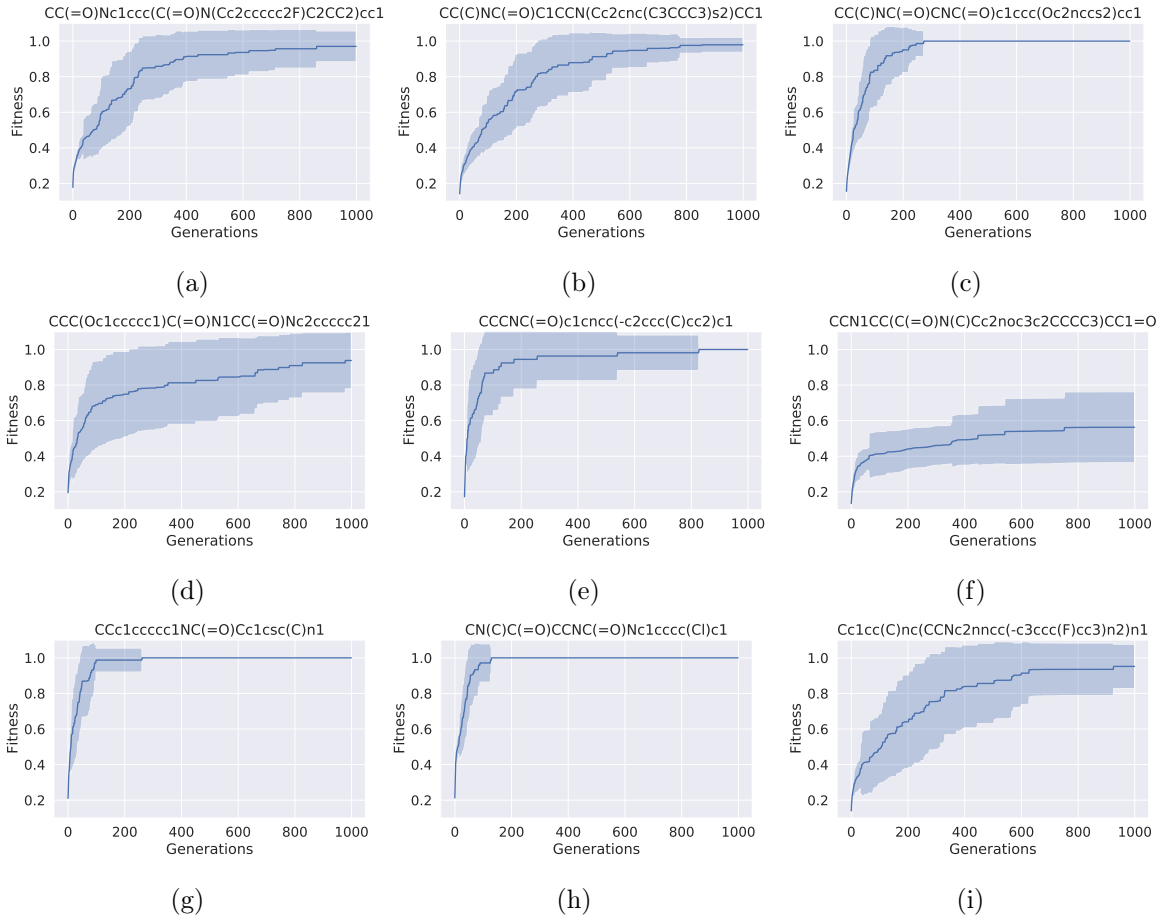


Figure 7.2: Fitness plots for nine different target molecules. Per molecule, 30 independent runs are executed, and the fitness of the best-performing individual of every generation is averaged over the runs. The shaded area marks plus-minus one standard deviation.

constructed with the same configuration and compared to the target using the Tanimoto similarity. All runs were terminated as soon as a molecule with a similarity to the target fingerprint of 1.0 has been evolved or after 1000 generations at the latest.

The results are presented in Figure 7.2. Even though the molecules came from the test set, all target molecules could theoretically be assembled from fragments in the fragment library. Throughout the 30 repetitions per target, the GA reconstructed the target molecule at least once for all nine targets. On average, 86% of all runs resulted in the target molecule being reconstructed. The fitness graphs indicate that the GA can continuously optimize molecules toward better-performing individuals. The molecules appear to have varying degrees of difficulty. Especially for the molecule pictured in Figure 7.2f, the algorithm struggled to generate molecules with a high similarity consistently. The fitness values in Figure 7.2d and Figure 7.2i also show higher standard deviations over the course of the generations compared to those of the other molecules. For the molecules in Figure 7.2c, 7.2g, and 7.2h, the GA succeeded in evolving molecules with a similarity to the target of 1.0 in the first few hundred generations in all 30 runs.

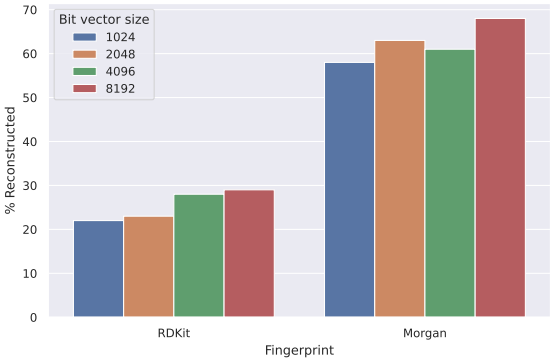


Figure 7.3: Percentage of reconstructed individuals for the RDKit and Morgan fingerprint with varying bit vector sizes.

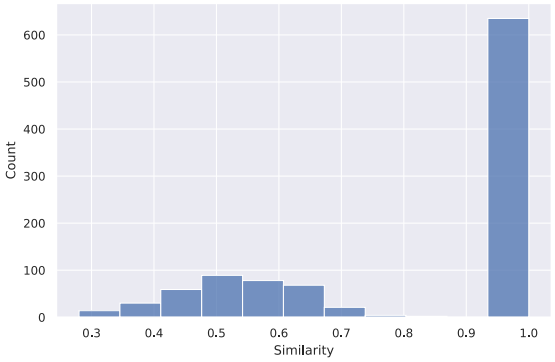


Figure 7.4: Frequency of Tanimoto similarities between the target fingerprint and the fingerprint of the best molecule found by the GA for 1000 different targets.

So far, we have used Morgan fingerprints with a bit vector size of 4096 for the fitness evaluation. However, Le, Winter, Noé, and Clevert [Le+20] reported different reconstruction results depending on the fingerprint's size. Furthermore, since it is conceivable that the way fingerprints encode information influences the structure of the fitness landscape, we compared the reconstruction performance for the topological RDKit fingerprint and the circular Morgan fingerprint. We executed independent runs for the two fingerprint types and four different bit vector sizes: $\{1024, 2048, 4096, 8192\}$. Every experimental condition was performed on 100 different random target molecules from the test set. For each target molecule, one GA run was executed per condition. Figure 7.3 shows how many of the 100 molecules could be reconstructed. A comparison of the two representations demonstrates an evident difference. The GA was able to reconstruct 22% to 29% of the topological RDKit fingerprints, whereas the circular Morgan fingerprint could be reconstructed in 58% to 68% of the cases. Comparing the bit vector sizes, there seems to be a slight advantage in the direction of larger bit vectors for both types of fingerprints. All in all, the results indicate that the reconstruction performance depends on the fingerprint representation chosen. The GA showed a higher success rate for reconstructing circular fingerprints. O'Boyle and Sayle [OS16] investigated how well different fingerprints capture molecule similarity and concluded that although the best performing fingerprint depends on the particular context, ECFPs are among the best at ranking diverse structures by similarity. It is conceivable that this leads to an easier-to-optimize fitness landscape.

To test the GA on a broader range of targets, we performed a GA run for 1000 different test molecules, using Morgan fingerprints with a bit vector size of 4096 for fitness evaluation. Of the 1000 molecules, 97.5% could theoretically be constructed from the set of BRICS fragments used. The results are pictured in Figure 7.4. The GA terminated with a similarity of 1.0 for 63.5% of the targets. There are only a few cases of final similarities in the range of 0.8 to 0.9, suggesting that the GA can optimize the molecules to fit perfectly if such a similar molecule

is found. There are cases in which the GA terminated with similarity scores around 0.5, which could indicate the existence of local optima in which the optimization is trapped. It is conceivable that GA would benefit from more advanced exploration methods. Although the molecules are not reconstructed entirely, they still demonstrate structural similarities to the target molecule. An exemplary overview of ten target molecules and their reconstruction is given in Figure 7.5.

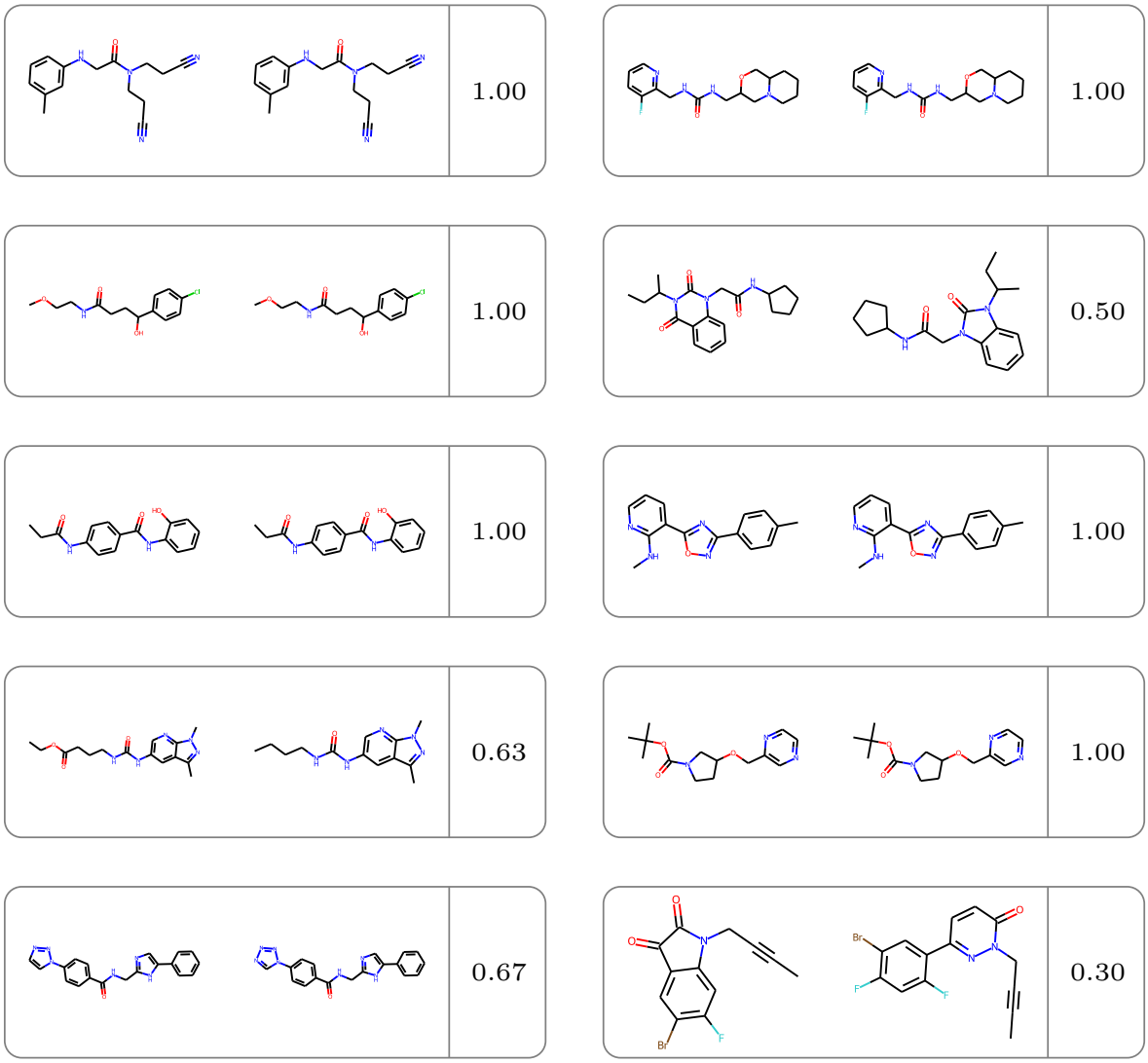


Figure 7.5: A randomly chosen selection of results from the reconstruction experiments. Every box pictures a target molecule (left) and the best reconstruction found by the GA (right). The number represents the Tanimoto similarity between the target and the reconstruction, where 1.00 means a perfect match between the fingerprints.

In summary, the results indicate that the proposed fragment-based GA is a suitable method for reconstructing molecular fingerprints. In the following section, we demonstrate how the GA can be incorporated into other procedures to create a molecular generation model.

7.2.2 Fingerprint Transformer

The previous experiments demonstrate the capabilities of the proposed GA for molecular fingerprint reconstruction. Combining the GA with other programs could enable them to operate directly on molecular fingerprints in the form of fixed-size bit vectors rather than on more complex representations such as graphs. Depending on the particular task, e.g., optimization or generation of molecules, the models can be designed to generate fingerprints that encode the desired properties without expending computational resources toward generating valid and realistic molecules. The GA maps the transformer-generated fingerprints to valid molecules composed of known molecular fragments.

As an exemplary use case, we investigated how a neural language model trained on molecular fingerprints can be applied to the task of generating new molecules. Molecular generative models are a common tool for identifying and designing new molecular structures, e.g., in the process of in silico drug design (see Section 3.3). For our study, a model was trained on a data set of real molecular fingerprints with the aim of capturing the underlying distribution of molecules that lead to these fingerprints. The model was then used to generate new fingerprints that should encode similar features to those present in the training data. Using the proposed GA, the newly generated fingerprints could then be decoded into novel molecules whose properties should also be similar to the training data.

Different types of generation models could be applied to this problem. We employed a deep neural network for fingerprint generation in the present study. The network was based on the transformer architecture [Vas+17]. Initially designed for language modeling tasks, this architecture is well suited for sequence processing as it features a powerful attention mechanism. Attention enables the model to focus on different points in a sequence without a need for recurrence, allowing it to combine information independent of its position in the given sequence. We employed an encoder based on this architecture which generates molecular fingerprints by processing them in an autoregressive manner—element for element—while considering previously generated elements. The model was trained on fingerprints generated from the MOSES training data set of ZINC molecules, with the objective of predicting the likelihood of a given fingerprint. A fingerprint's likelihood $p(\mathbf{f})$ can be factorized in the conditional probability:

$$p(\mathbf{f}) = \prod_{i=1}^{N} p(f_i | f_{<i}) \tag{7.1}$$

where N is the number of elements in the fingerprint and $p(f_i|f_{<i})$ is the probability of the occurrence of an element at position i given the previous elements in the fingerprint. We utilized PyTorch's standard transformer implementation for the given task and constructed the model from six stacked transformer encoder layers. The dimensionality of the feature vectors was configured to 512, and the model used eight attention heads. For training, the molecules were converted to Morgan fingerprints using an atom radius of six and a bit vector size of 2048. To decrease computation time and reduce the input sequence length, the fingerprints were split into chunks of size eight so that each chunk contained one byte. This led to a

vocabulary of 256 tokens plus one artificial token marking the beginning of each fingerprint. Since fingerprints are fixed in size, no padding or *end of sequence* tokens were required. Before the sequences were fed into the transformer, they were processed by a learnable embedding layer, and a positional encoding was applied. The model was trained with a cross entropy loss, and the Adam optimizer was used with a learning rate of 10^{-4}.

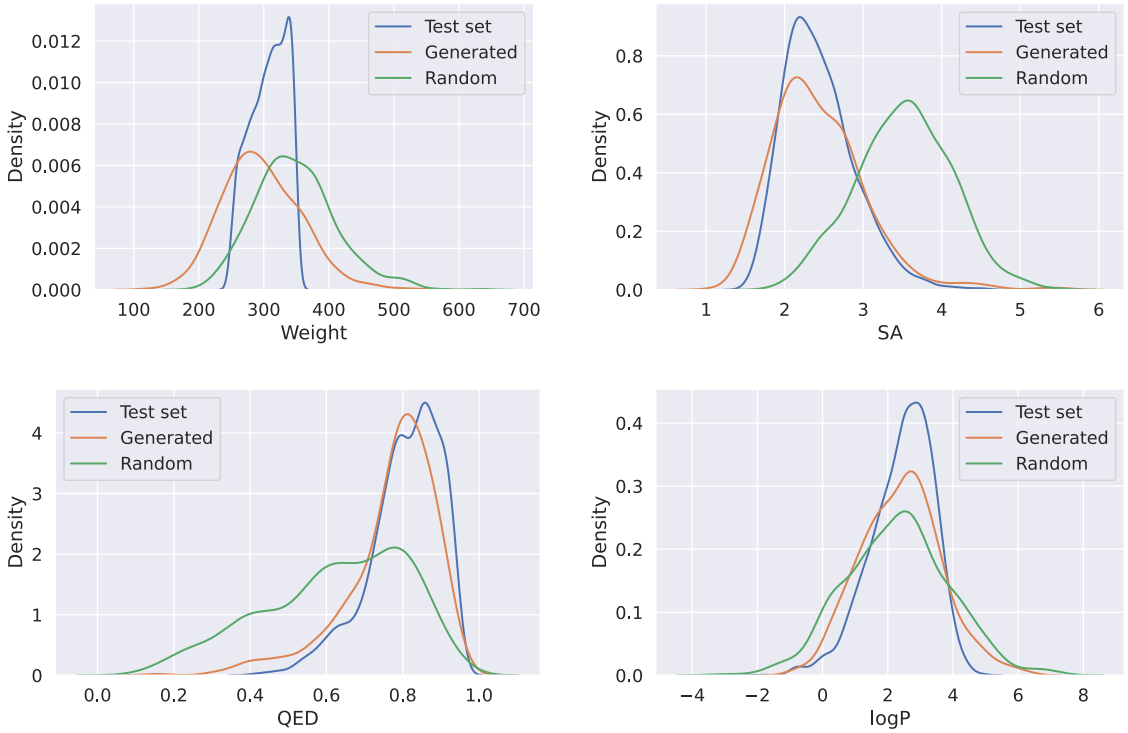
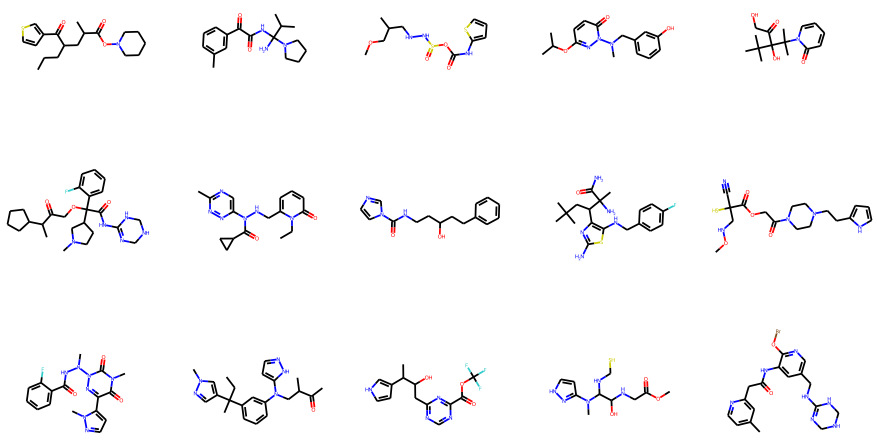
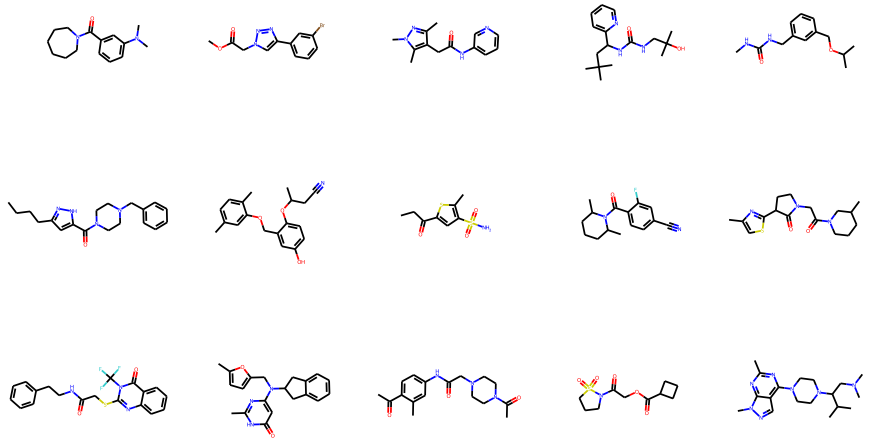


Figure 7.6: Distributions of molecular metrics Weight, SA, QED, and logP in test data, the molecules reconstructed from random fingerprints, and the molecules reconstructed from fingerprints generated by the transformer.

We generated 500 fingerprints by initializing them with the start token and sampling them in an autoregressive manner. To analyze the quality of the generated fingerprints, a baseline of randomly generated fingerprints was created. The random fingerprints were constructed by inferring from the training fingerprints the probability of a given index in the fingerprint being one. Each element of the fingerprint was drawn from a Bernoulli distribution parameterized with these probabilities. We used the GA to decode 500 transformer-generated and 500 random fingerprints to molecules for 1000 generations. Since the transformer was trained on molecules from the MOSES ZINC data set, the fingerprints generated should resemble those from the training data, and therefore the GA should be able to decode them into molecules with similar properties compared to the molecules in ZINC. The random fingerprints should, on the other hand, be more complicated to decode and differ from the original molecules. All in all, the molecules constructed as designed by the transformer showed a mean Tanimoto similarity of 0.4 to their respective fingerprint, whereas the random baseline could be reconstructed to molecules with an average similarity of 0.22. Both models generated only



(a) Decoded random fingerprints



(b) Decoded transformer-generated fingerprints

Figure 7.7: Reconstruction of (a) randomly and (b) transformer-generated fingerprints.

valid molecules. All molecules proposed by the transformer resulted in unique molecules, and 90.86% of the molecules were novel compared to the MOSES data set. For the decoded random fingerprint, 99.41% of the molecules were unique, and 100% were new. To investigate how well these molecules resemble the training data, we picture the distribution of four different molecular properties in Figure 7.6. The four properties, also included in MOSES, are the molecular weight, the SA score, a heuristic for qualifying how easy a molecule is to synthesize, the QED estimating how suitable a molecule would be as a drug candidate, and the water-octanol partition coefficient which measures how well a molecule mixes with water [Pol+20]. As a reference, we compare the distributions for these properties between the molecules generated by the random fingerprints, the transformer fingerprints, and the molecules included in the test set of ZINC molecules that were not used for training. When considering the distribution plots, it is evident that the molecules generated from the random

fingerprints differ from the test data in these properties, although they are composed of similar fragments. Most noticeably, they exhibit an on average lower QED and a worse SA. For molecules reconstructed from the transformer-generated fingerprints, the distribution of logP and molecular weights indicate a higher spread for the generated molecules when compared to the test data. One possible explanation could be that the reconstructed fingerprints are bit vectors that do not encode the frequency of a substructure in the molecule and therefore provide no direct incentive to reconstruct molecules with the exact number of atoms. However, the molecules perform better than the baseline regarding drug-likeness and synthetic accessibility. The distributions of these metrics clearly resemble the corresponding test data distributions. It is conceivable that the chosen fingerprint captures these more holistic molecular properties more easily than more concrete features, such as the exact number of atoms. For one, this indicates the transformer's capability to describe new molecules with similar characteristics to the training data. Furthermore, it demonstrates that the proposed GA can decode molecules in a way that matches the specifications of the respective fingerprint. Since the GA allows for the simple inclusion of additional fitness metrics or other fingerprint representations, it could be promising to explore whether they allow specifying different target properties of the generated molecules. An exemplary overview of the generated molecule is given in Figure 7.7.

All in all, the property distributions demonstrate that the transformer is able to generate fingerprints that encode meaningful information about the training data and which relate to new but similar molecules. The experiments show how the introduced GA enables other approaches to work directly with fingerprint representations allowing the specification of desired molecule properties, while the GA handles the task of decoding the fingerprints to valid molecules.

7.3 Conclusion

In this study, we present a GA capable of combining molecular fragments to generate molecules and optimize them to resemble target fingerprints. The algorithm can be employed to reconstruct molecules based on their fingerprint representation, for which the circular Morgan fingerprint seems particularly suitable. Working with molecule fragments rather than on an atom basis ensures that the resulting molecules are valid and contain realistic substructures. To construct a diverse set of retrosynthetically interesting fragments, we employ the BRICS algorithm on molecules from the ZINC library of drug-like molecules. Using a set of molecule fragments allows control over the generated molecules, as it enables the inclusion or exclusion of substructures depending on the specific problem. Furthermore, we demonstrate how the GA opens up the easily accessible fingerprint representation as the basis for other molecular generation or optimization models by utilizing a fingerprint transformer to discover new molecules. In contrast to other methods based on deep learning, applying the GA to new problem domains does not require re-training and opens up the possibility of incorporating additional fitness functions or constraints into the process.

Future work could include adapting the workflow for other molecular optimization problems, such as drug discovery. In this context, using fingerprints containing information about pharmacophoric features may be particularly well suited, as it could help encode the molecule properties responsible for biological activity. Applying the approach to the reconstruction of continuous molecular descriptors rather than bit vectors, as described by Winter, Montanari, Noé, and Clevert [Win+19], could be a promising follow-up. It is conceivable that these continuous descriptors provide a different fitness landscape more suitable for optimization.

Part III

Conclusion and Future Work

8 Conclusion

This thesis aims to improve AI-based molecule generation models for the four molecule representations most commonly used in this field. To achieve this, we leverage different AI techniques—namely optimization algorithms and deep generative learning—over the course of four consecutive research studies. We introduce new generation models based on genetic algorithms for multi-objective design and transformers for molecule generation of sequential representations. Both AI techniques offer unique features and advantages, as they are quite different approaches to molecule generation. Combined, the results indicate that more than a single AI method is needed to address the variety of problems that molecular design presents. There is no one best strategy for molecule generation. Likewise, there are a variety of ways to represent the molecules. And for them, too, this thesis demonstrates that each offers unique advantages but also challenges in terms of generation. In addition, the combination of the representation and generation technique provides a variety of opportunities to exploit the strengths of these approaches fully.

In what follows, we will discuss these findings in detail by summarizing and combining the results of the presented work. Furthermore, we will explore what recommendations can be derived from this study and what possible future work could arise from it.

8.1 Representations

Throughout the experiments in this thesis, we have seen applications of molecule generation based on the four major molecular representations utilized in de novo molecule generation: string, graph, spatial, and fingerprint representations. The obtained results not only reinforce the knowledge about their advantages and disadvantages but also provide new insights into their applicability for molecule generation. This section summarizes and combines these findings.

String-based representations offer an easy entry point into molecule generation, as demonstrated in the Chapter 4. There are huge libraries of druglike molecules in string representations available. Most chemical computation frameworks offer straightforward implementations for processing molecular strings. These factors and the simple sequential nature of strings have led to various approaches based on this representation. Although the SMILES representation is most common, it is not particularly suitable for representing individuals in GAs. Since random alterations of a SMILES string most likely result in invalid molecules, such a GA would require complex mutation operators to ensure validity. In our study on the design of SARS-CoV 2 inhibitors, we demonstrate that the SELFIES representation, on the

other hand, can easily be evolved with a GA. The SELFIES syntax ensures molecule validity, even for a completely random string. This property makes the algorithm independent of any knowledge about chemical rules and molecular structures and allows applying a standard GA approach to the problem. For example, in our work, we apply simple yet effective mutation operators, like point mutations, to alter the SELFIES. In comparison, mutation operators for SMILES would have to take syntactic and chemical rules into consideration to ensure feasible molecules.

We show that the GA based on individuals represented as SELFIES is able to navigate through the vast search space of potential molecules. In this process, the selected genetic operations are capable of generating a wide variety of molecular structures. However, the experiments also demonstrate the disadvantages of the SELFIES representation. Since parsing a SELFIES ends when the valence of a molecule is satisfied, even if symbols remain in the string, random mutations are biased toward decreasing the molecules' size. This complicates evolving larger molecules and the implementation of crossover operators.

Lessons Learned – Strings

With its validity enforcing syntax, SELFIES offers a straightforward representation for GAs.

While the simplicity of string representations is an advantage when it comes to accessibility, it sometimes misappropriates information that is relevant to the task at hand. Therefore, in another study, we investigate three-dimensional representations that encode the entire spatial information about a molecule. Such a representation does not need abstract concepts like bonds or rings and allows the description of spatial isomers of the same molecule (see Section 3.1). In particular, for problems with known structural requirements, a three-dimensional representation can include this information in its search. However, constructing a suitable three-dimensional representation—especially for molecule generation—is a difficult task.

In our study, we decided against a voxelized representation, as it has disadvantages that can not be dismissed. Embedding molecules in voxel spaces leaves most of the space empty, resulting in significant computational effort for subsequent operations. Furthermore, the molecules are not invariant to rotation, which renders training of machine learning methods less efficient. Based on the Euclidean distance matrix, our generation approach allows us to work directly with atom positions and is invariant to rotation and translation. A representation based on Euclidean distances requires a further post-processing step to approximate the respective set of coordinates. We utilize multidimensional scaling in our approach, which is a sound strategy, as shown by the results of the molecular dynamics analysis. It is also conceivable to extend the approach to generate an atom's position directly, for example, similar to E(n)-equivariant graph neural networks [SHW21].

The generally smaller number of valid molecules in our results suggests that the generation of molecules in 3D is more complicated than with string representations. This is to be expected, as more chemical knowledge has to be captured by a spatial model. Furthermore, a

spatial model introduces additional sources of errors into the generation. It is not sufficient to state that two elements of a molecule are connected. Adding a valid connecting component requires specifying appropriate distances and angles to already generated atoms. Finally, it should be noted that the storage and conversion of molecules into other representations becomes more complicated when three-dimensional information is involved.

Despite these drawbacks, our approach shows significant advantages of including spatial information in the generation process. Due to the training procedure featuring varying atom permutations for every molecule, our approach can complete unfinished Euclidean distance matrices to valid molecules. This demonstrates that it is possible to incorporate desired structural properties into the generation process. An advantage that could be particularly useful in drug design where a respective target structure is known. A three-dimensional representation allows selecting promising substructures and placing them at the desired spatial positions to ensure that the generated molecule meets the spatial requirements of the problem. All in all, spatial representations promise significant advantages over other representations. However, generating molecules in 3D is considerably more complicated and introduces additional challenges, which is probably why there are hardly any approaches using this representation. There is great potential for the further development of such algorithms.

Lessons Learned – Spatial Representations

Three-dimensional representations allow molecules to be directly tailored to the spatial requirements of the problem. Yet, they introduce additional challenges for molecule generation.

Using a representation based on Euclidean distances allows a straightforward transfer of the presented approach toward graph generation. Our approach in Chapter 6 represents and generates molecular graphs via the adjacency matrix. This sets it apart from other state-of-the-art graph generation approaches, e.g., the common message passing neural networks. However, this design translates graph generation to a sequence generation problem—in our case, the prediction of adjacency matrix columns—which allows applying language processing techniques like transformers.

In general, encoding and generating molecules in the form of graphs has several advantages. Unlike string-based representations, graphs directly express an atom's connections and the corresponding bond types. For strings, this information may be scattered throughout the string and must be inferred based on the syntactic rules. Adding an atom to a molecule is also easier since only the desired bonds need to be specified. In the case of strings, such an extension may require additional symbols, for example, to indicate opening branches or ring structures. In the case of a spatial representation, the new atom would have to be placed in precisely the correct position to create these bonds. This is also reflected in our experiments by the generally high number of valid molecules generated, despite the fact that the molecules were significantly larger than in the spatial generation experiment. Overall, with the graph

representation introduced, our algorithm was able to generate a variety of different molecules with similar properties to the training data.

One of the disadvantages of the presented approach is that, as with strings, spatial information is not represented. A further disadvantage is a need to decide on an order of the molecules' atoms, which is usually not inherent. However, this also opens up possibilities for further research comparing different orders. For some applications, it is conceivable that a particular generation order is advantageous, e.g., starting at outer points and growing toward the center.

Lessons Learned – Graphs

The representation of molecules as graphs via the adjacency matrix enables the application of sequential generation strategies, such as transformers. The simplicity of graphs, as opposed to spatial representations, allows larger molecules to be efficiently sampled.

Molecular fingerprints are an uncommon representation for molecule generation. Much more frequently, this representation is applied to similarity search in virtual screening. Due to their fixed size and discrete value ranges, fingerprints are one of the simplest representations imaginable regarding processing and storage. Especially for generation, a fixed output size is advantageous to many generation models, e.g., neural networks.

In our study presented in Chapter 7, we demonstrate how—with the help of a GA for fingerprint reconstruction—fingerprints can indeed be used for molecule generation. This way, we can transfer sequence generation techniques from language processing to the generation of fingerprints and, building on this, to the generation of molecules. This poses a different generation paradigm compared to the other de novo approaches. The generation model does not directly propose new molecules but rather a molecular descriptor that specifies the respective desired molecular characteristics. A subsequent procedure is then used to construct realistic molecules based on these descriptors. Our results indicate that the final molecules indeed show the predefined characteristics, as they show similar property distributions to the molecules used to train the fingerprint generation model.

Due to the involvement of two different procedures, the overall approach is more complicated than the other introduced strategies. However, this also opens up the possibility of combining the advantages of multiple approaches. In our case, the fast and data-driven generation of fingerprints with the flexible, fragment-based GA. It is also conceivable to use other molecular descriptors, for example, those containing pharmacological information, to tailor the generation to the specific problem area.

Lessons Learned – Fingerprints

Due to their non-decodability, molecular fingerprints are usually not used for molecule generation. However, fingerprints become a valuable, straightforward representation for generation models when combined with a reconstruction GA.

8.2 Genetic Algorithms and Transformers

In addition to different representations, we utilize two distinct AI methods for the design of molecules in this thesis, namely GAs and transformers. These methods, too, have a unique profile of advantages and disadvantages with respect to the various applications in molecular design. Moreover, the usefulness of these approaches depends on the representation chosen and the specific problem requirements. In this section, we will highlight key findings and compare the two different strategies.

Inhibitor development is a complex process, and compounds must be designed with respect to multiple target metrics. GAs have a rich theoretical foundation for multi-objective design. In Chapter 4, we demonstrate how GAs can be used for the multi-objective design of SARS-CoV 2 inhibitors and illustrate the GA's capability of generating a diverse set of promising candidate molecules. In particular, when combined with NSGA-II, the algorithm generated a diverse set of candidate molecules distributed across the Pareto front. Due to the usage of a GA, the approach is flexible to changes in the objectives or the target protease. Unlike a neural network-based approach, which requires complete retraining or parameter refinement, a GA can be restarted directly if conditions change.

The study on the design of SARS-CoV 2 inhibitors also reveals disadvantages of GAs. Due to their heuristic nature, GAs lack an intrinsic motivation for generating realistic molecules. Realism must be enforced via additional objectives, but these are often evaluated with heuristics that can be exploited.

Lessons Learned – Genetic Algorithms

GAs are a powerful tool for the generation of protease inhibitors, as they can consider the multi-objective nature of this problem domain. However, GAs lack an intrinsic motivation for generating realistic, drug-like molecules.

A further advantage of GA is their flexibility in terms of the possible molecule representations. In the case of the fingerprint generation study presented in Chapter 7, the use of a GA allows for a flexible representation based on trees of molecular fragments. The representation can be used to generate molecules of arbitrary size and enables a variety of possible mutation and crossover operators. Another example is the SELFIES representation, chosen in the first study. GA can easily be applied to this representation, while it is conceivable that the ambiguous meaning of SELFIES tokens could pose a problem for neural networks.

Molecule generation with transformers, as an instance of deep generative models, involves a different design philosophy. In contrast to GAs, this approach is not necessarily centered around optimization. Instead, the models are data-driven and trained to generate molecules that could originate from the distribution underlying the training data. Training these models can take considerable time, but new molecules can be sampled nearly instantaneously.

As demonstrated in the studies on three-dimensional and graph-based molecule generation (Chapter 5 and Chapter 6), transformers are a powerful tool for generating realistic molecules. Once a suitable sequential representation is found, transformers can learn the complex

relationships underlying molecular composition simply by observation. Remarkably, this also holds true for the complex chemical principles influencing spatial molecule generation. Our experiments demonstrate that the predicted conformation of the generated molecules is close to the results predicted by the molecular dynamics simulations. It is conceivable that the powerful attention mechanism, in particular, enables the transformer to capture these relations. Due to this technique, the transformer can easily combine information at different positions of the input sequence to decide on the next sampling step. In contrast, a GA would require some form of conformation estimating heuristic to generate three-dimensional molecules. However, such an approach would only be as good as the accuracy of these heuristics, and the actual molecules would likely exhibit different conformations.

Compared to GAs, deep generative models are not as flexible regarding the structure of inputs and outputs. Sequential sampling is required to allow the generation of molecules with an arbitrary number of atoms. This is straightforwardly applicable to string-based representations, but spatial generation presents additional challenges. Typically, sequential models are trained with the goal of increasing the likelihood of a sequence sampled from categorical distributions. This is more difficult to model for continuous three-dimensional atom spaces. In our design, we overcame this challenge by discretizing the distance space into bins.

Lessons Learned – Transformers

By sequentializing the Euclidean distance matrix or adjacency matrix of a molecule, transformers can be used as generation tools for three-dimensional molecules and molecular graphs. Their unique attention mechanism helps transformers to capture even complex chemical relationships.

The study presented in Chapter 7 demonstrates how GAs and transformers can be combined into one molecule generation model. This way, the approach utilizes the advantages of both techniques. A transformer approximates a probability distribution underlying the given set of molecules and allows the sampling of new and realistic descriptors. The flexibility of the GA allows molecules to be optimized to fit these descriptors while using a flexible representation, ensuring all molecules are valid and consist of retrosynthetically interesting fragments. All in all, these results demonstrate that various methods have their justification when it comes to generating molecules and that sometimes even a combination of methods can be advantageous.

8.3 Future Work

Although some suggestions for extending the presented approaches are already included in the research sections, this section concludes with additional general guidance for future work. Overall, it is conceivable to transfer the lessons learned from the latter studies to real-world drug design problems. The first study already gives an idea of how GAs can be used to solve present, highly relevant problems in molecule design. In particular, incorporating spatial

information into the process could be a promising area for further research. Binding to a target protease imposes structural requirements on drug design, and a spatial approach could incorporate this information directly into its search.

However, further research is needed to improve the standing of spatial representations in molecular design. Due to their simplicity and availability, research mainly focuses on string-based representations. There is great potential for developing new spatial representations invariant to rotation, translation, and permutation yet allowing for simple training and sampling procedures.

The design of generative models constructing molecules from molecule descriptors, as demonstrated in the Chapter 7, also shows great potential. Such a design paradigm has multiple advantages. It decouples the design of a molecular property profile from the actual structural design of a molecule on an atomic basis. Our study shows that each task can be approached with an individual, customized technique. This can make it easier to use different strategies, adapt them to the respective tasks and combine their distinct advantages. It is also conceivable that the overall interpretability of generative models in molecular design could be improved by utilizing interpretable descriptors that include task-specific information, e.g., pharmacological information. Moreover, practitioners could interact with the generation model as they can review and improve the proposed descriptors.

In conclusion, the presented studies introduce new ways to utilize AI strategies for generating molecules in different representations. Our findings have the potential to advance the role of AI in molecular design to facilitate humanity's search for innovative materials and medicine.

References

[Alh+15] A. Alhossary, S. D. Handoko, Y. Mu, and C. K. Kwoh. "Fast, accurate, and reliable molecular docking with QuickVina 2". In: *Bioinformatics* 31.13 (2015), pp. 2214–2216. ISSN: 14602059. DOI: 10.1093/bioinformatics/btv082.

[All+80] N. L. Allinger, M. P. Cava, D. C. de Jongh, C. R. Johnson, N. A. Lebel, C. L. Stevens, and M. P. Cava. *Organische Chemie*. Ed. by G. Koßmehl. De Gruyter, Dec. 1980. ISBN: 978-3-11-004594-9. DOI: 10.1515/9783110829709.

[Baj02] J. Bajorath. "Integration of virtual and high-throughput screening". In: *Nature Reviews Drug Discovery* 1.11 (Nov. 2002), pp. 882–894. ISSN: 1474-1776. DOI: 10.1038/nrd941.

[BCB15] D. Bahdanau, K. Cho, and Y. Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2015. URL: http://arxiv.org/abs/1409.0473.

[Bec93] A. D. Becke. "Density-functional thermochemistry. III. The role of exact exchange". In: *The Journal of Chemical Physics* 98.7 (Apr. 1993). ISSN: 0021-9606. DOI: 10.1063/1.464913.

[Bic+12] G. R. Bickerton, G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins. "Quantifying the chemical beauty of drugs". In: *Nature Chemistry* 4.2 (2012), pp. 90–98. ISSN: 17554330. DOI: 10.1038/nchem.1243.

[BNE07] N. Beume, B. Naujoks, and M. Emmerich. "SMS-EMOA: Multiobjective selection based on dominated hypervolume". In: *European Journal of Operational Research* 181.3 (Sept. 2007), pp. 1653–1669. ISSN: 03772217. DOI: 10.1016/j.ejor.2006.08.008.

[Bro+04] N. Brown, B. McKay, F. Gilardoni, and J. Gasteiger. "A Graph-Based Genetic Algorithm and Its Application to the Multiobjective Evolution of Median Molecules". In: *Journal of Chemical Information and Computer Sciences* 44.3 (May 2004), pp. 1079–1087. ISSN: 0095-2338. DOI: 10.1021/ci034290p.

[Bro+19] N. Brown, M. Fiscato, M. H. Segler, and A. C. Vaucher. "GuacaMol: Benchmarking Models for de Novo Molecular Design". In: *Journal of Chemical Information and Modeling* 59.3 (Mar. 2019), pp. 1096–1108. ISSN: 1549-9596. DOI: 10.1021/acs.jcim.8b00839.

[Bro+20a] N. Brown, P. Ertl, R. Lewis, T. Luksch, D. Reker, and N. Schneider. "Artificial intelligence in chemistry and drug design". In: *Journal of Computer-Aided Molecular Design* 34.7 (July 2020), pp. 709–715. ISSN: 0920-654X. DOI: 10.1007/s10822-020-00317-x.

[Bro+20b] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. "Language Models are Few-Shot Learners". In: (2020). arXiv: 2005.14165 [cs.CL].

[But+18] K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, and A. Walsh. "Machine learning for molecular and materials science". In: *Nature* 559.7715 (July 2018). ISSN: 0028-0836. DOI: 10.1038/s41586-018-0337-2.

[Cal+20] L. Caly, J. D. Druce, M. G. Catton, D. A. Jans, and K. M. Wagstaff. "The FDA-approved drug ivermectin inhibits the replication of SARS-CoV-2 in vitro". In: *Antiviral Research* 178 (June 2020), p. 104787. ISSN: 01663542. DOI: 10.1016/j.antiviral.2020.104787.

[CBJ19] B. Chen, R. Barzilay, and T. S. Jaakkola. "Path-Augmented Graph Transformer Network". In: *CoRR* abs/1905.1 (2019). arXiv: 1905.12712. URL: http://arxiv.org/abs/1905.12712.

[Che+20] G. Chen, Z. Shen, A. Iyer, U. F. Ghumman, S. Tang, J. Bi, W. Chen, and Y. Li. "Machine-Learning-Assisted De Novo Design of Organic Molecules and Polymers: Opportunities and Challenges". In: *Polymers* 12.1 (Jan. 2020), p. 163. ISSN: 2073-4360. DOI: 10.3390/polym12010163. URL: https://www.mdpi.com/2073-4360/12/1/163.

[CK21] T. Cofala and O. Kramer. "Transformers for Molecular Graph Generation". In: *ESANN 2021 proceedings*. Louvain-la-Neuve (Belgium): Ciaco - i6doc.com, 2021, pp. 123–128. ISBN: 978287587082-7. DOI: 10.14428/esann/2021.ES2021-112.

[CK22] T. Cofala and O. Kramer. "An evolutionary fragment-based approach to molecular fingerprint reconstruction". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, July 2022, pp. 1156–1163. ISBN: 9781450392372. DOI: 10.1145/3512290.3528824.

[CL20] D. Cai and W. Lam. "Graph Transformer for Graph-to-Sequence Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.05 (Apr. 2020), pp. 7464–7471. ISSN: 2374-3468. DOI: 10.1609/aaai.v34i05.6243. arXiv: 1911.07470.

[Cly22] A. Clyde. "Ultrahigh Throughput Protein–Ligand DockingDockingwith Deep Learning". In: *Artificial Intelligence in Drug Design*. Ed. by A. Heifetz. New York, NY: Springer US, 2022, pp. 301–319. ISBN: 978-1-0716-1787-8. DOI: 10.1007/978-1-0716-1787-8_13.

[Cof+20] T. Cofala, L. Elend, P. Mirbach, J. Prellberg, T. Teusch, and O. Kramer. "Evolutionary Multi-objective Design of SARS-CoV-2 Protease Inhibitor Candidates". In: *Parallel Problem Solving from Nature - PPSN XVI - 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part II*. Ed. by T. Bäck, M. Preuss, A. H. Deutz, H. Wang, C. Doerr, M. T. M. Emmerich, and H. Trautmann. Vol. 12270. Lecture Notes in Computer Science. Springer, 2020, pp. 357–371. DOI: 10.1007/978-3-030-58115-2_25.

[CPH06] J. A. Costa, N. Patwari, and A. O. Hero. "Distributed weighted-multidimensional scaling for node localization in sensor networks". In: *ACM Transactions on Sensor Networks* 2.1 (Feb. 2006). ISSN: 1550-4859. DOI: 10.1145/1138127.1138129.

[CTK21] T. Cofala, T. Teusch, and O. Kramer. "Spatial Generation of Molecules with Transformers". In: *International Joint Conference on Neural Networks*. IEEE, July 2021, pp. 1–7. ISBN: 978-1-6654-3900-8. DOI: 10.1109/IJCNN52387.2021.9533439. © 2021 IEEE.

[Dai+20] W. Dai, B. Zhang, X.-M. Jiang, H. Su, J. Li, Y. Zhao, X. Xie, Z. Jin, J. Peng, F. Liu, C. Li, Y. Li, F. Bai, H. Wang, X. Cheng, X. Cen, S. Hu, X. Yang, J. Wang, X. Liu, G. Xiao, H. Jiang, Z. Rao, L.-K. Zhang, Y. Xu, H. Yang, and H. Liu. "Structure-based design of antiviral drug candidates targeting the SARS-CoV-2 main protease". In: *Science* 368.6497 (June 2020), pp. 1331–1335. ISSN: 0036-8075. DOI: 10.1126/science.abb4489.

[Deg+08] J. Degen, C. Wegscheid-Gerlach, A. Zaliani, and M. Rarey. "On the Art of Compiling and Using 'Drug-Like' Chemical Fragment Spaces". In: *ChemMedChem* 3.10 (Oct. 2008), pp. 1503–1507. ISSN: 18607179. DOI: 10.1002/cmdc.200800178.

[Dev+19] J. Devlin, M. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by J. Burstein, C. Doran, and T. Solorio. Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/n19-1423.

[Dha+20] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever. "Jukebox: A Generative Model for Music". In: *CoRR* abs/2005.00341 (2020). arXiv: 2005.00341.

[DHP71] R. Ditchfield, W. J. Hehre, and J. A. Pople. "Self-Consistent Molecular-Orbital Methods. IX. An Extended Gaussian-Type Basis for Molecular-Orbital Studies of Organic Molecules". In: *The Journal of Chemical Physics* 54.2 (Jan. 1971). ISSN: 0021-9606. DOI: 10.1063/1.1674902.

[DSC15] R. V. Devi, S. S. Sathya, and M. S. Coumar. "Evolutionary algorithms for de novo drug design – A survey". In: *Applied Soft Computing* 27 (Feb. 2015), pp. 543–552. ISSN: 15684946. DOI: 10.1016/j.asoc.2014.09.042.

[DTG00] D. Douguet, E. Thoreau, and G. Grassy. "A Genetic Algorithm for the Automated Generation of Small Organic Molecules: Drug Design Using an Evolutionary Algorithm". In: *Journal of Computer-Aided Molecular Design* 14.5 (2000), pp. 449–466. ISSN: 0920654X. DOI: 10.1023/A:1008108423895.

[ERS08] P. Ertl, S. Roggo, and A. Schuffenhauer. "Natural Product-likeness Score and Its Application for Prioritization of Compound Libraries". In: *Journal of Chemical Information and Modeling* 48.1 (Jan. 2008), pp. 68–74. ISSN: 1549-9596. DOI: 10.1021/ci700286x.

[ES09] P. Ertl and A. Schuffenhauer. "Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions". In: *Journal of Cheminformatics* 1.1 (Dec. 2009), p. 8. ISSN: 1758-2946. DOI: 10.1186/1758-2946-1-8.

[Fis+20] A. Fischer, M. Sellner, S. Neranjan, M. A. Lill, and M. Smieško. "Inhibitors for Novel Coronavirus Protease Identified by Virtual Screening of 687 Million Compounds". In: *ChemRxiv. Preprint.* (2020), pp. 1–21. DOI: 10.26434/CHEMRXIV.11923239.V1.

[Fos19] D. Foster. *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play.* O'Reilly Media, 2019, p. 385. ISBN: 978-1-492-04194-8.

[FOW66] L. Fogel, A. Owens, and M. Walsh. *Artificial Intelligence Through Simulated Evolution.* Wiley, 1966. ISBN: 9780471265160.

[FPB84] M. J. Frisch, J. A. Pople, and J. S. Binkley. "Self-consistent molecular orbital methods 25. Supplementary functions for Gaussian basis sets". In: *The Journal of Chemical Physics* 80.7 (Apr. 1984). ISSN: 0021-9606. DOI: 10.1063/1.447079.

[FS17] R. Ferreira de Freitas and M. Schapira. "A systematic analysis of atomic protein–ligand interactions in the PDB". In: *MedChemComm* 8.10 (2017), pp. 1970–1981. ISSN: 2040-2503. DOI: 10.1039/C7MD00381A.

[Gai18] T. Gaillard. "Evaluation of AutoDock and AutoDock Vina on the CASF-2013 Benchmark". In: *Journal of Chemical Information and Modeling* 58.8 (Aug. 2018), pp. 1697–1706. ISSN: 1549-9596. DOI: 10.1021/acs.jcim.8b00312.

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.

[GGS19] N. W. A. Gebauer, M. Gastegger, and K. Schütt. "Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett. 2019, pp. 7564–7576. URL: https://proceedings.neurips.cc/paper/2019/hash/a4d8e2a7e0d0c102339f97716d2fdfb6-Abstract.html.

[GMH22] C. Grebner, H. Matter, and G. Hessler. "Artificial Intelligence in Compound Design". In: *Artificial Intelligence in Drug Design*. Ed. by A. Heifetz. New York, NY: Springer US, 2022, pp. 349–382. ISBN: 978-1-0716-1787-8. DOI: 10.1007/978-1-0716-1787-8_15.

[Gom+17] J. Gomes, B. Ramsundar, E. N. Feinberg, and V. S. Pande. "Atomic Convolutional Networks for Predicting Protein-Ligand Binding Affinity". In: *CoRR* abs/1703.10603 (2017). arXiv: 1703.10603.

[Góm+18] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik. "Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules". In: *ACS Central Science* 4.2 (2018). ISSN: 2374-7943. DOI: 10.1021/acscentsci.7b00572.

[GS22] F. Grisoni and G. Schneider. "De Novo Molecular Design with Chemical Language Models". In: *Artificial Intelligence in Drug Design*. Ed. by A. Heifetz. New York, NY: Springer US, 2022, pp. 207–232. ISBN: 978-1-0716-1787-8. DOI: 10.1007/978-1-0716-1787-8_9.

[Ham08] K. Hamberg. "Gender Bias in Medicine". In: *Women's Health* 4.3 (May 2008), pp. 237–243. ISSN: 1745-5065. DOI: 10.2217/17455057.4.3.237.

[Han+12] S. D. Handoko, X. Ouyang, C. T. T. Su, C. K. Kwoh, and Y. Ong. "QuickVina: Accelerating AutoDock Vina Using Gradient-Based Heuristics for Global Optimization". In: *IEEE ACM Trans. Comput. Biol. Bioinform.* 9.5 (2012), pp. 1266–1272. DOI: 10.1109/TCBB.2012.82.

[HDP72] W. J. Hehre, R. Ditchfield, and J. A. Pople. "Self—Consistent Molecular Orbital Methods. XII. Further Extensions of Gaussian—Type Basis Sets for Use in Molecular Orbital Studies of Organic Molecules". In: *The Journal of Chemical Physics* 56.5 (Mar. 1972). ISSN: 0021-9606. DOI: 10.1063/1.1677527.

[Heu18] L. Heuer. "AI could threaten pharmaceutical patents". In: *Nature* 558.7711 (June 2018), pp. 519–519. ISSN: 0028-0836. DOI: 10.1038/d41586-018-05555-6.

[HN19] M. Hoffmann and F. Noé. "Generating valid Euclidean distance matrices". In: *CoRR* abs/1910.03131 (2019). arXiv: 1910.03131.

[Hol75] J. H. Holland. "Adaptation in natural and artificial systems". In: *The University of Michigan Press* (1975).

[Hor+12] E. van der Horst, P. Marqués-Gallego, T. Mulder-Krieger, J. van Veldhoven, J. Kruisselbrink, A. Aleman, M. T. M. Emmerich, J. Brussee, A. Bender, and A. P. IJzerman. "Multi-Objective Evolutionary Design of Adenosine Receptor Ligands". In: *Journal of Chemical Information and Modeling* 52.7 (July 2012), pp. 1713–1721. ISSN: 1549-9596. DOI: 10.1021/ci2005115.

[HT20] T. Hey and A. Trefethen. "The Fourth Paradigm 10 Years On". In: *Informatik Spektrum* 42.6 (Jan. 2020), pp. 441–447. ISSN: 0170-6012. DOI: 10.1007/s00287-019-01215-9.

[HTT09] T. Hey, S. Tansley, and K. Tolle. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, Oct. 2009. ISBN: 978-0-9825442-0-4. URL: https://www.microsoft.com/en-us/research/publication/fourth-paradigm-data-intensive-scientific-discovery/.

[Imr+20] F. Imrie, A. R. Bradley, M. van der Schaar, and C. M. Deane. "Deep Generative Models for 3D Linker Design". In: *Journal of Chemical Information and Modeling* 60.4 (Apr. 2020). ISSN: 1549-9596. DOI: 10.1021/acs.jcim.9b01120.

[JBJ18] W. Jin, R. Barzilay, and T. S. Jaakkola. "Junction Tree Variational Autoencoder for Molecular Graph Generation". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by J. G. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 2328–2337. URL: http://proceedings.mlr.press/v80/jin18a.html.

[Jen19] J. H. Jensen. "A graph-based genetic algorithm and generative model/Monte Carlo tree search for the exploration of chemical space". In: *Chemical Science* 10.12 (2019), pp. 3567–3572. ISSN: 2041-6520. DOI: 10.1039/C8SC05372C.

[Jin+20] Z. Jin, X. Du, Y. Xu, Y. Deng, M. Liu, Y. Zhao, B. Zhang, X. Li, L. Zhang, C. Peng, Y. Duan, J. Yu, L. Wang, K. Yang, F. Liu, R. Jiang, X. Yang, T. You, X. Liu, X. Yang, F. Bai, H. Liu, X. Liu, L. W. Guddat, W. Xu, G. Xiao, C. Qin, Z. Shi, H. Jiang, Z. Rao, and H. Yang. "Structure of Mpro from SARS-CoV-2 and discovery of its inhibitors". In: *Nature* 582.7811 (June 2020), pp. 289–293. ISSN: 0028-0836. DOI: 10.1038/s41586-020-2223-y.

[Jum+21] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis. "Highly accurate protein

structure prediction with AlphaFold". In: *Nature* 596.7873 (2021), pp. 583–589. ISSN: 14764687. DOI: 10.1038/s41586-021-03819-2.

[KC18] R. D. King and P. Courtney. "Dilemma over AI and drug patenting already under debate". In: *Nature* 560.7718 (Aug. 2018), pp. 307–307. ISSN: 0028-0836. DOI: 10.1038/d41586-018-05955-8.

[KH05] S. S. Kaplan and C. B. Hicks. "Safety and antiviral activity of lopinavir/ ritonavir-based therapy in human immunodeficiency virus type 1 (HIV-1) infection". In: *Journal of Antimicrobial Chemotherapy* 56.2 (Aug. 2005), pp. 273–276. ISSN: 1460-2091. DOI: 10.1093/jac/dki209.

[Kim+21] S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang, and E. E. Bolton. "PubChem in 2021: new data content and improved web interfaces". In: *Nucleic Acids Research* 49.D1 (Jan. 2021), pp. D1388–D1395. ISSN: 0305-1048. DOI: 10.1093/nar/gkaa971.

[KP20] M. Klimek and M. Perelstein. "Neural network-based approach to phase space integration". In: *SciPost Physics* 9.4 (Oct. 2020), p. 053. ISSN: 2542-4653. DOI: 10.21468/SciPostPhys.9.4.053.

[Kra+18] T. Krause, N. Röckendorf, N. El-Sourani, K. Ramaker, M. Henkel, S. Hauke, M. Borschbach, and A. Frey. "Breeding Cell Penetrating Peptides: Optimization of Cellular Uptake by a Function-Driven Evolutionary Process". In: *Bioconjugate Chemistry* 29.12 (Dec. 2018), pp. 4020–4029. ISSN: 1043-1802. DOI: 10.1021/acs.bioconjchem.8b00583.

[Kra17] O. Kramer. *Genetic Algorithm Essentials*. Springer International Publishing, 2017, p. 92. ISBN: 9783642285851. DOI: 10.1007/978-3-319-52156-5.

[Kre+20] M. Krenn, F. Häse, A. Nigam, P. Friederich, and A. Aspuru-Guzik. "Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation". In: *Mach. Learn. Sci. Technol.* 1.4 (2020), p. 45024. DOI: 10.1088/2632-2153/aba947.

[Kri+80] R. Krishnan, J. S. Binkley, R. Seeger, and J. A. Pople. "Self-consistent molecular orbital methods. XX. A basis set for correlated wave functions". In: *The Journal of Chemical Physics* 72.1 (Jan. 1980). ISSN: 0021-9606. DOI: 10.1063/1.438955.

[Kuz+18] D. Kuzminykh, D. Polykovskiy, A. Kadurin, A. Zhebrak, I. Baskov, S. Nikolenko, R. Shayakhmetov, and A. Zhavoronkov. "3D Molecular Representations Based on the Wave Transform for Convolutional Neural Networks". In: *Molecular Pharmaceutics* 15.10 (Oct. 2018). ISSN: 1543-8384. DOI: 10.1021/acs.molpharmaceut.7b01134.

[Kwo+21] Y. Kwon, S. Kang, Y.-S. Choi, and I. Kim. "Evolutionary design of molecules based on deep learning and a genetic algorithm". In: *Scientific Reports* 11.1 (Dec. 2021), p. 17304. ISSN: 2045-2322. DOI: 10.1038/s41598-021-96812-8.

[Lam+06] E.-W. Lameijer, J. N. Kok, T. Bäck, and A. P. IJzerman. "The Molecule Evoluator. An Interactive Evolutionary Algorithm for the Design of Drug-Like Molecules". In: *Journal of Chemical Information and Modeling* 46.2 (Mar. 2006), pp. 545–552. ISSN: 1549-9596. DOI: 10.1021/ci050369d.

[Le+20] T. Le, R. Winter, F. Noé, and D.-A. Clevert. "Neuraldecipher – reverse-engineering extended-connectivity fingerprints (ECFPs) to their molecular structures". In: *Chemical Science* 11.38 (2020), pp. 10378–10389. ISSN: 2041-6520. DOI: 10.1039/D0SC03115A.

[Li+18] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. W. Battaglia. "Learning Deep Generative Models of Graphs". In: *CoRR* abs/1803.03324 (2018). arXiv: 1803.03324.

[Lia+19] R. Liao, Y. Li, Y. Song, S. Wang, W. Hamilton, D. K. Duvenaud, R. Urtasun, and R. Zemel. "Efficient Graph Generation with Graph Recurrent Attention Networks". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019.

[Lip+97] C. A. Lipinski, F. Lombardo, B. W. Dominy, and P. J. Feeney. "Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings". In: *Advanced Drug Delivery Reviews* 23.1-3 (Jan. 1997), pp. 3–25. ISSN: 0169409X. DOI: 10.1016/S0169-409X(96)00423-1.

[Liu21] X. Liu. *Organic Chemistry I*. Kwantlen Polytechnic University, 2021. ISBN: 978-1-989864-52-4.

[LYP88] C. Lee, W. Yang, and R. G. Parr. "Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density". In: *Physical Review B* 37.2 (Jan. 1988). ISSN: 0163-1829. DOI: 10.1103/PhysRevB.37.785.

[Ma+15] J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, and V. Svetnik. "Deep Neural Nets as a Method for Quantitative Structure–Activity Relationships". In: *Journal of Chemical Information and Modeling* 55.2 (Feb. 2015), pp. 263–274. ISSN: 1549-9596. DOI: 10.1021/ci500747n.

[Mas+08] B. B. Masek, L. Shen, K. M. Smith, and R. S. Pearlman. "Sharing chemical information without sharing chemical structure". In: *Journal of chemical information and modeling* 48.2 (2008), pp. 256–261.

[Men+19] D. Mendez, A. Gaulton, A. P. Bento, J. Chambers, M. De Veij, E. Félix, M. P. Magariños, J. F. Mosquera, P. Mutowo, M. Nowotka, M. Gordillo-Marañón, F. Hunter, L. Junco, G. Mugumbate, M. Rodriguez-Lopez, F. Atkinson, N. Bosc, C. J. Radoux, A. Segura-Cabrera, A. Hersey, and A. R. Leach. "ChEMBL: towards direct deposition of bioassay data". In: *Nucleic Acids Research* 47.D1 (Jan. 2019), pp. D930–D940. ISSN: 0305-1048. DOI: 10.1093/nar/gky1075.

[Mir19] S. Mirjalili. “Genetic Algorithm”. In: *Evolutionary Algorithms and Neural Networks: Theory and Applications*. Cham: Springer International Publishing, 2019, pp. 43–55. ISBN: 978-3-319-93025-1. DOI: 10.1007/978-3-319-93025-1_4.

[MKR16] T. Mueller, A. G. Kusne, and R. Ramprasad. “Machine Learning in Materials Science”. In: *Reviews in Computational Chemistry*. John Wiley & Sons, Ltd, 2016. Chap. 4, pp. 186–273. ISBN: 9781119148739. DOI: https://doi.org/10.1002/9781119148739.ch4.

[Mor+09] G. M. Morris, R. Huey, W. Lindstrom, M. F. Sanner, R. K. Belew, D. S. Goodsell, and A. J. Olson. “AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility”. In: *Journal of Computational Chemistry* 30.16 (Dec. 2009), pp. 2785–2791. ISSN: 01928651. DOI: 10.1002/jcc.21256. URL: https://onlinelibrary.wiley.com/doi/10.1002/jcc.21256.

[MP20] E. K. McCreary and J. M. Pogue. “Coronavirus Disease 2019 Treatment: A Review of Early and Emerging Options”. In: *Open Forum Infectious Diseases* 7.4 (Apr. 2020). ISSN: 2328-8957. DOI: 10.1093/ofid/ofaa105.

[MPP20] M. Macchiagodena, M. Pagliai, and P. Procacci. “Inhibition of the Main Protease 3CL-pro of the Coronavirus Disease 19 via Structure-Based Ligand Design and Molecular Modeling”. In: *arXiv:2002.09937 [q-bio]* (Mar. 2020). arXiv: 2002.09937 [q-bio].

[MRD22] C. Muller, O. Rabal, and C. Diaz Gonzalez. “Artificial Intelligence, Machine Learning, and Deep Learning in Real-Life Drug Design Cases”. In: *Artificial Intelligence in Drug Design*. Ed. by A. Heifetz. New York, NY: Springer US, 2022, pp. 383–407. ISBN: 978-1-0716-1787-8. DOI: 10.1007/978-1-0716-1787-8_16.

[NB13] C. A. Nicolaou and N. Brown. “Multi-objective optimization methods in drug design”. In: *Drug Discovery Today: Technologies* 10.3 (Sept. 2013), e427–e435. ISSN: 17406749. DOI: 10.1016/j.ddtec.2013.02.001.

[Nee12] F. Neese. “The ORCA program system”. In: *WIREs Computational Molecular Science* 2.1 (Jan. 2012). ISSN: 1759-0876. DOI: 10.1002/wcms.81.

[Nee18] F. Neese. “Software update: the ORCA program system, version 4.0”. In: *WIREs Computational Molecular Science* 8.1 (Jan. 2018). ISSN: 1759-0876. DOI: 10.1002/wcms.1327.

[Nig+20] A. Nigam, P. Friederich, M. Krenn, and A. Aspuru-Guzik. “Augmenting Genetic Algorithms with Deep Neural Networks for Exploring the Chemical Space”. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL: https://openreview.net/forum?id=H1lmyRNFvr.

[OS16] N. M. O’Boyle and R. A. Sayle. “Comparing structural fingerprints using a literature-based similarity benchmark”. In: *Journal of Cheminformatics* 8.1 (Dec. 2016), p. 36. ISSN: 1758-2946. DOI: 10.1186/s13321-016-0148-0.

[Pau+10] S. M. Paul, D. S. Mytelka, C. T. Dunwiddie, C. C. Persinger, B. H. Munos, S. R. Lindborg, and A. L. Schacht. "How to improve R&D productivity: the pharmaceutical industry's grand challenge". In: *Nature Reviews Drug Discovery* 9.3 (Mar. 2010), pp. 203–214. ISSN: 1474-1776. DOI: 10.1038/nrd3078.

[PBM20] M. Podda, D. Bacciu, and A. Micheli. "A Deep Generative Model for Fragment-Based Molecule Generation". In: *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*. Ed. by S. Chiappa and R. Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 2020, pp. 2240–2250. URL: http://proceedings.mlr.press/v108/podda20a.html.

[Pet96] A. Petrowski. "A clearing procedure as a niching method for genetic algorithms". In: *Proceedings of IEEE International Conference on Evolutionary Computation ICEC-96*. IEEE. IEEE, 1996, pp. 798–803. ISBN: 0-7803-2902-3. DOI: 10.1109/ICEC.1996.542703.

[PHK01] S. C.-H. Pegg, J. J. Haresco, and I. D. Kuntz. "A genetic algorithm for structure-based de novo design". In: *Journal of Computer-Aided Molecular Design* 15.10 (2001), pp. 911–933. ISSN: 0920654X. DOI: 10.1023/A:1014389729000.

[Pol+20] D. Polykovskiy, A. Zhebrak, B. Sanchez-Lengeling, S. Golovanov, O. Tatanov, S. Belyaev, R. Kurbanov, A. Artamonov, V. Aladinskiy, M. Veselov, A. Kadurin, S. Johansson, H. Chen, S. Nikolenko, A. Aspuru-Guzik, and A. Zhavoronkov. "Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models". In: *Frontiers in Pharmacology* 11 (Dec. 2020), p. 565644. ISSN: 1663-9812. DOI: 10.3389/fphar.2020.565644. eprint: 1811.12823.

[Rad+19] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. "Language Models are Unsupervised Multitask Learners". Unpublished. 2019.

[Ram+14] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. Von Lilienfeld. "Quantum chemistry structures and properties of 134 kilo molecules". In: *Scientific Data* 1 (2014), pp. 1–7. ISSN: 20524463. DOI: 10.1038/sdata.2014.22.

[Ram+22] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. "Hierarchical Text-Conditional Image Generation with CLIP Latents". In: *CoRR* abs/2204.06125 (2022). DOI: 10.48550/arXiv.2204.06125.

[RBF12] N. Röckendorf, M. Borschbach, and A. Frey. "Molecular Evolution of Peptide Ligands with Custom-Tailored Characteristics for Targeting of Glycostructures". In: *PLoS Computational Biology* 8.12 (Dec. 2012). Ed. by Y. Ofran, e1002800. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1002800.

[RCW15] A. M. Rush, S. Chopra, and J. Weston. "A Neural Attention Model for Abstractive Sentence Summarization". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2015, pp. 379–389. DOI: 10.18653/v1/D15-1044.

[Rec73] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Problemata (Stuttgart). Frommann-Holzboog, 1973. ISBN: 9783772803741.

[RH10] D. Rogers and M. Hahn. "Extended-Connectivity Fingerprints". In: *Journal of Chemical Information and Modeling* 50.5 (May 2010), pp. 742–754. ISSN: 1549-9596. DOI: 10.1021/ci100050t.

[RL13] S. Riniker and G. A. Landrum. "Open-source platform to benchmark fingerprints for ligand-based virtual screening". In: *Journal of Cheminformatics* 5.1 (Dec. 2013), p. 26. ISSN: 1758-2946. DOI: 10.1186/1758-2946-5-26.

[RN16] S. Russell and P. Norvig. *Artificial intelligence : a modern approach*. Boston Columbus Indianapolis New York San Francisco, 2016.

[Roc+16] T. Rocktäschel, E. Grefenstette, K. M. Hermann, T. Kociský, and P. Blunsom. "Reasoning about Entailment with Neural Attention". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2016. URL: http://arxiv.org/abs/1509.06664.

[Ron+20] Y. Rong, Y. Bian, T. Xu, W. Xie, Y. WEI, W. Huang, and J. Huang. "Self-Supervised Graph Transformer on Large-Scale Molecular Data". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 12559–12571. URL: https://proceedings.neurips.cc/paper/2020/hash/94aef38441efa3380a3bed3faf1f9d5d-Abstract.html.

[RS94] G. Rudolph and H.-P. Schwefel. "Evolutionäre Algorithmen - ein robustes Optimierkonzept". In: *Physik Journal* 50.3 (Mar. 1994), pp. 236–238. ISSN: 00319279. DOI: 10.1002/phbl.19940500309.

[Rud+12] L. Ruddigkeit, R. Van Deursen, L. C. Blum, and J. L. Reymond. "Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17". In: *Journal of Chemical Information and Modeling* 52.11 (2012), pp. 2864–2875. ISSN: 15499596. DOI: 10.1021/ci300415d.

[San+21] D. Santos-Martins, L. Solis-Vasquez, A. F. Tillack, M. F. Sanner, A. Koch, and S. Forli. "Accelerating AutoDock 4 with GPUs and Gradient-Based Local Search". In: *Journal of Chemical Theory and Computation* 17.2 (Feb. 2021), pp. 1060–1073. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.0c01006.

[SC19] G. Schneider and D. E. Clark. "Automated De Novo Drug Design: Are We Nearly There Yet?" In: *Angewandte Chemie International Edition* 58.32 (Aug. 2019), pp. 10792–10803. ISSN: 1433-7851. DOI: 10.1002/anie.201814681.

[Sch+20] P. Schneider, W. P. Walters, A. T. Plowright, N. Sieroka, J. Listgarten, R. A. Goodnow, J. Fisher, J. M. Jansen, J. S. Duca, T. S. Rush, M. Zentgraf, J. E. Hill, E. Krutoholow, M. Kohler, J. Blaney, K. Funatsu, C. Luebkemann, and G. Schneider. "Rethinking drug design in the artificial intelligence era". In: *Nature*

Reviews Drug Discovery 19.5 (May 2020). ISSN: 1474-1776. DOI: 10.1038/s41573-019-0050-3.

[Sch18] G. Schneider. "Automating drug discovery". In: *Nature Reviews Drug Discovery* 17.2 (Feb. 2018), pp. 97–113. ISSN: 1474-1776. DOI: 10.1038/nrd.2017.232.

[Sch77] H.-P. Schwefel. *Numerische Optimierung von Computermodellen mittels der Evolutionsstrategie*. Vol. 26. Birkhäuser Basel, Jan. 1977. ISBN: 9783764308766. DOI: 10.1007/978-3-0348-5927-1.

[Seg+18] M. H. S. Segler, T. Kogej, C. Tyrchan, and M. P. Waller. "Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks". In: *ACS Central Science* 4.1 (2018), pp. 120–131. DOI: 10.1021/acscentsci.7b00512.

[SG20] D. Schwalbe-Koda and R. Gómez-Bombarelli. "Generative Models for Automatic Chemical Design". In: *Machine Learning Meets Quantum Physics*. Ed. by K. T. Schütt, S. Chmiela, O. A. von Lilienfeld, A. Tkatchenko, K. Tsuda, and K.-R. Müller. Cham: Springer International Publishing, 2020, pp. 445–467. ISBN: 978-3-030-40245-7. DOI: 10.1007/978-3-030-40245-7_21.

[SHW21] V. G. Satorras, E. Hoogeboom, and M. Welling. "E(n) Equivariant Graph Neural Networks". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 9323–9332.

[SI15] T. Sterling and J. J. Irwin. "ZINC 15 – Ligand Discovery for Everyone". In: *Journal of Chemical Information and Modeling* 55.11 (Nov. 2015), pp. 2324–2337. ISSN: 1549-9596. DOI: 10.1021/acs.jcim.5b00559.

[SPW18] M. H. S. Segler, M. Preuss, and M. P. Waller. "Planning chemical syntheses with deep neural networks and symbolic AI". In: *Nature* 555.7698 (Mar. 2018), pp. 604–610. ISSN: 0028-0836. DOI: 10.1038/nature25978.

[Tam+20] A. Tamashiro, T. Yoshio, A. Ishiyama, T. Tsuchida, K. Hijikata, S. Yoshimizu, Y. Horiuchi, T. Hirasawa, A. Seto, T. Sasaki, J. Fujisaki, and T. Tada. "Artificial intelligence-based detection of pharyngeal cancer using convolutional neural networks". In: *Digestive Endoscopy* (2020), pp. 1057–1065. ISSN: 14431661. DOI: 10.1111/den.13653.

[Tho+22] M. Thomas, A. Boardman, M. Garcia-Ortegon, H. Yang, C. de Graaf, and A. Bender. "Applications of Artificial Intelligence in Drug Design: Opportunities and Challenges". In: *Artificial Intelligence in Drug Design*. Ed. by A. Heifetz. New York, NY: Springer US, 2022, pp. 1–59. ISBN: 978-1-0716-1787-8. DOI: 10.1007/978-1-0716-1787-8_1.

[TO09] O. Trott and A. J. Olson. "AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading". In: *Journal of Computational Chemistry* (2009), NA–NA. ISSN: 01928651. DOI: 10.1002/jcc.21334.

[Urb+22] F. Urbina, F. Lentzos, C. Invernizzi, and S. Ekins. "Dual use of artificial-intelligence-powered drug discovery". In: *Nature Machine Intelligence* 4.3 (Mar. 2022), pp. 189–191. ISSN: 2522-5839. DOI: 10.1038/s42256-022-00465-9.

[Vas+17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 6000–6010.

[Vig19] J. Vig. "A Multiscale Visualization of Attention in the Transformer Model". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2019, pp. 37–42. DOI: 10.18653/v1/P19-3007.

[Wag+16] T. T. Wager, X. Hou, P. R. Verhoest, and A. Villalobos. "Central Nervous System Multiparameter Optimization Desirability: Application in Drug Discovery". In: *ACS Chemical Neuroscience* 7.6 (June 2016), pp. 767–775. ISSN: 1948-7193. DOI: 10.1021/acschemneuro.6b00029.

[WDH15] I. Wallach, M. Dzamba, and A. Heifets. "AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery". In: *CoRR* abs/1510.02855 (2015). URL: http://arxiv.org/abs/1510.02855.

[Wei88] D. Weininger. "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules". In: *Journal of Chemical Information and Modeling* 28.1 (Feb. 1988). ISSN: 1549-9596. DOI: 10.1021/ci00057a005.

[WGL00] R. Wang, Y. Gao, and L. Lai. "LigBuilder: A Multi-Purpose Program for Structure-Based Drug Design". In: *Journal of Molecular Modeling* 6.7-8 (Aug. 2000), pp. 498–516. ISSN: 1610-2940. DOI: 10.1007/s0089400060498.

[Win+19] R. Winter, F. Montanari, F. Noé, and D.-A. Clevert. "Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations". In: *Chemical Science* 10.6 (2019), pp. 1692–1701. ISSN: 2041-6520. DOI: 10.1039/C8SC04175J.

[Wol+20] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. "Transformers: State-of-the-Art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2020, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6.

[Xu22] Y. Xu. "Deep Neural Networks for QSAR". In: *Artificial Intelligence in Drug Design*. Ed. by A. Heifetz. New York, NY: Springer US, 2022, pp. 233–260. ISBN: 978-1-0716-1787-8. DOI: 10.1007/978-1-0716-1787-8_10.

[Xue+19] D. Xue, Y. Gong, Z. Yang, G. Chuai, S. Qu, A. Shen, J. Yu, and Q. Liu. "Advances and challenges in deep generative models for de novo molecule generation". In: *WIREs Computational Molecular Science* 9.3 (May 2019). ISSN: 1759-0876. DOI: 10.1002/wcms.1395.

[Yoo+20] S. Yoo, Y.-S. Kim, K. H. Lee, K. Jeong, J. Choi, H. Lee, and Y. S. Choi. "Graph-Aware Transformer: Is Attention All Graphs Need?" In: *CoRR* abs/2006.0 (2020). arXiv: 2006.05213. URL: https://arxiv.org/abs/2006.05213.

[You+18] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec. "GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by J. G. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 5694–5703. URL: http://proceedings.mlr.press/v80/you18a.html.

[YPL20] Y. Yuan, J. Pei, and L. Lai. "LigBuilder V3: A Multi-Target de novo Drug Design Approach". In: *Frontiers in Chemistry* 8 (Feb. 2020). ISSN: 2296-2646. DOI: 10.3389/fchem.2020.00142.

[Zha+19] A. Zhavoronkov, Y. A. Ivanenkov, A. Aliper, M. S. Veselov, V. A. Aladinskiy, A. V. Aladinskaya, V. A. Terentiev, D. A. Polykovskiy, M. D. Kuznetsov, A. Asadulaev, Y. Volkov, A. Zholus, R. R. Shayakhmetov, A. Zhebrak, L. I. Minaeva, B. A. Zagribelnyy, L. H. Lee, R. Soll, D. Madge, L. Xing, T. Guo, and A. Aspuru-Guzik. "Deep learning enables rapid identification of potent DDR1 kinase inhibitors". In: *Nature Biotechnology* 37.9 (Sept. 2019), pp. 1038–1040. ISSN: 1087-0156. DOI: 10.1038/s41587-019-0224-x.

[Zha+20] L. Zhang, D. Lin, X. Sun, U. Curth, C. Drosten, L. Sauerhering, S. Becker, K. Rox, and R. Hilgenfeld. "Crystal structure of SARS-CoV-2 main protease provides a basis for design of improved α-ketoamide inhibitors". In: *Science* 368.6489 (Apr. 2020), pp. 409–412. ISSN: 0036-8075. DOI: 10.1126/science.abb3405.

List of Figures

1.1 Integration of in silico molecule design in the DMTA cycle. 5
1.2 Four common types of molecular representation, illustrated using an exemplary organic molecule. 6
1.3 Structure of this thesis. 8

2.1 Objective landscapes of optimization problems. 14
2.2 Utilizing a generative model for data synthesis. 17

3.1 Exemplary organic molecules. 20
3.2 A molecular graph with one of the possible SMILES strings and the canonical SMILES representation. 22
3.3 The molecule vanillin in two different three-dimensional representations. . . . 23
3.4 Navigation in the multimodal molecule landscape with the help of a deep generative model. This model embeds molecules in the two-dimensional design space. Starting from an origin molecule (red circle), the model can be used to explore the objective space by sampling similar but new molecules in the design space. 29
3.5 The two most promising DDR1 inhibitors found by Zhavoronkov et al. with the help of deep learning. 31

4.1 Illustration of a protease enzyme with uncut precursor polyproteins, the cleavage progress, and protease inhibition preventing the cleavage. 40
4.2 Molecular structure formula, SMILES, and SELFIES of 2-fluorophenol. 44
4.3 Development of all metrics during single-objective and multi-objective NSGA-II optimization runs. 48
4.4 Typical Pareto fronts evolved with NSGA-II: (a) docking score vs. QED, (b) docking score vs. NP, (c) docking score vs. SA. 48
4.5 Comparison of the last generation of an exemplary single-objective and NGSA-II run. 50
4.6 Exemplary protease inhibitors with properties presented as radar plot, structural formula, and chemical name. 51
4.7 Molecules PI-I and PI-V docked to the pocket of SARS-CoV-2's M^{pro}. 52

5.1 Overview of the transformer architecture. 55
5.2 Structure of the training data, illustrated by an exemplary molecule. 62

5.3 Comparison between the QM9 database and the two experimental conditions on the average number of atoms and rings per molecule. 65
5.4 Comparison of radial and angular distribution functions between QM9 and the generated molecules. 66
5.5 Results of the completion experiment. 67

6.1 Overview of the model's architecture. 71

7.1 The BRICS fragments for an exemplary molecule. 81
7.2 Fitness plots for ten different target molecules. 84
7.3 Percentage of reconstructed individuals for the RDKit and Morgan fingerprint with varying bit vector sizes. 85
7.4 Frequency of Tanimoto similarities between the target fingerprint and the fingerprint of the best molecule found by the GA for 1000 different targets. . . . 85
7.5 A randomly chosen selection of results from the reconstruction experiments. . 86
7.6 Distributions of molecular metrics Weight, SA, QED, and logP in test data, the molecules reconstructed from random fingerprints, and the molecules reconstructed from fingerprints generated by the transformer. 88
7.7 Reconstruction of randomly and transformer-generated fingerprints. 89

List of Tables

4.1 Value ranges and optima for the employed metrics. 42
4.2 Experimental results of the best molecules generated by the single-objective approach and NSGA-II. 49

5.1 Results of the generation procedure for the two experimental conditions, grouped according to the quality of the molecules. 64

6.1 Comparison of molecules created by different generative models. 73
6.2 Similarities between generated molecules and the test/scaffold test set. 74

7.1 Statistics of data set and resulting fragment library. 81

Eidesstattlichen Versicherung

Hiermit versichere ich an Eides statt, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Außerdem versichere ich, dass ich die allgemeinen Prinzipien wissenschaftlicher Arbeit und Veröffentlichung, wie sie in den Leitlinien guter wissenschaftlicher Praxis der Carl von Ossietzky Universität Oldenburg festgelegt sind, befolgt habe.

Oldenburg, den 8. Juni 2023

Tim Cofala