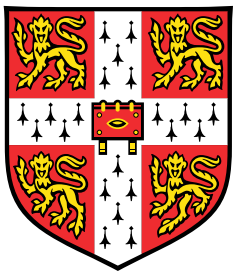
Generative Models for Synthesizable Lipids



Jingyi Zhao

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy in Machine Learning and Machine Intelligence

Robinson College August 2024

To all the beautiful things and kind people in my life.

Declaration

I, Jingyi Zhao of Robinson College, being a candidate for the Master of Philosophy in Machine Learning and Machine Intelligence, hereby declare that this dissertation and the work described in it are my own work, unaided except as specified below, and that this dissertation does not contain material that has already been used for a comparable purpose.
Software Declaration: This project utilized the following work, together with standard Python libraries and PyTorch framework:

1. **Lipid Analyzer** for lipid tail extraction: this is an unpublished work developed by Shahab Sepehri, source codes are not publicly available.
2. **MolGpKa**[1] for pKa prediction (Pan et al., 2021)
3. **SyntheMol**[2] for the foundation of our baseline naive Monte Carlo tree search method for molecular generation (Swanson et al., 2024)
4. **Chemprop**[3] for the framework that we built the lipid classifier upon (Yang et al., 2019)
5. **Syntheseus**[4] for retrosynthesis planning

Our main work is split into the following two GitHub repositories:

1. **Lipid Building Block Dataset Construction**: this is collaborated with Yuxuan Ou. https://github.com/yuxuanou623/Lipid_reaction_dataset_creation
2. **MCTS for Lipid Generation**: containing the naive MCTS method built upon the SyntheMol framework and the guided MCTS approach implemented by myself. https://github.com/Stella-zjy/SyntheMol-Lipid

Word Count: 13856

Jingyi Zhao
August 2024

[1] https://github.com/Xundrug/MolGpKa
[2] https://github.com/swansonk14/SyntheMol
[3] https://github.com/chemprop/chemprop
[4] https://github.com/microsoft/syntheseus

Acknowledgements

First and foremost, I would like to acknowledge my supervisors, José Miguel Hernández-Lobato and Morteza Rasoulianboroujeni, for proposing the central idea upon which this project is based and for their guidance and support throughout this project.

I would like to acknowledge Austin Tripp, who helped advise me on this project, for sharing his insight and knowledge and for his huge support over the past four months.

I would like to acknowledge Asal Mehradfar and Mohammad Shahab Sepehri for their preceding work on this project. Especially for sharing the lipid dataset used for training lipid classifier and for sharing the Lipid Analyzer toolkit.

I would also like to acknowledge Yuxuan Ou, my dearest friend and classmate, for collaborating with me on the dataset creation process and for her companion and support during the past year.

Last but certainly not least, I would like to acknowledge all MLMI classmates, faculty, and staff, and University of Cambridge for my master year.

Abstract

Messenger RNA (mRNA) has emerged as a crucial tool in biomedicine, making significant advancements in the treatment and prevention of diverse diseases. The inherent instability of mRNA necessitates a robust delivery mechanism, primarily provided by ionizable lipid nanoparticles (LNPs) (Kularatne et al., 2022; Mitchell et al., 2021). These LNPs consist of four distinct lipid types, among which the ionizable amine-containing lipids are pivotal. LNPs with unique ionizable lipid structures are tailored to deliver mRNA to specific target cells and tissues, underscoring the critical need for ongoing innovation in the design of ionizable lipids to optimize mRNA-based therapeutics.

Deep generative models have emerged as a powerful solution to the inverse molecular design challenge, enabling the translation of desired molecular properties into specific molecular structures (Kusner et al., 2017; Liu et al., 2019; Simonovsky and Komodakis, 2018). However, a practical challenge arises as the molecular structures generated can often be difficult or infeasible to synthesize (Gao and Coley, 2020). Thus, integrating synthesis planning into the design of generative models is crucial.

In this project, we explore Monte Carlo tree search (MCTS)-based generative models for the generation of ionizable lipids. We begin by constructing a dataset of purchasable lipid building blocks, and develop two lipid property predictors—a lipid classifier and an ionizability predictor—to assess whether candidate molecules are lipid-like or ionizable. These predictors guide the MCTS in navigating the vast combinatorial chemical space comprised of the building block dataset and a template-based reaction prediction model. We introduce a policy network guided MCTS generative model capable of producing high-quality ionizable lipids with available synthesis paths. Our achievements indicate that this project offers a promising direction for ionizable lipid generation, also contributing to the broader field of drug delivery.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Overview and Motivation

The development of messenger RNA (mRNA)-based therapeutics marks a transformative advance in the treatment and prevention of a wide range of diseases, including genetic disorders, infectious diseases, and cancer (Kong et al., 2023; Sahin et al., 2014). Given the intrinsic instability of mRNA, a robust delivery mechanism is crucial, with ionizable lipid nanoparticles (LNPs) emerging as the leading technology for this purpose (Kularatne et al., 2022; Mitchell et al., 2021). These LNPs comprise four distinct lipid types, among which the ionizable amine-containing lipids are pivotal. They primarily facilitate the encapsulation of mRNA within LNPs and enhance its delivery into the cellular cytoplasm, where it can be translated into therapeutic proteins (Degors et al., 2019; Kim et al., 2021; Wittrup et al., 2015; Xu et al., 2021). Notably, different LNPs often feature unique ionizable lipid structures, emphasizing the importance of developing a diverse array of ionizable lipids. This diversity is crucial for effectively delivering mRNA to various target cells and tissues, highlighting the need for continued innovation in the design of ionizable lipids to facilitate mRNA-based therapeutics.

Designing novel ionizable lipids is traditionally time-consuming and labor-intensive. However, recent advancements in the integration of deep learning with combinatorial chemistry have shown great promise in accelerating this development (Xu et al., 2024). One of the primary challenges in ionizable lipid generation is the effective exploration of a vast combinatorial chemical space. The SyntheMol approach has demonstrated considerable success in this area, utilizing Monte Carlo tree search (MCTS) to efficiently navigate through expansive search spaces and generate desired molecules, complete with synthesis pathways, particularly in the field of antibiotic development (Swanson et al., 2024).

In this dissertation, we extend the application of the MCTS-based generative model to the domain of ionizable lipid generation. Building on the foundation established by SyntheMol, we have enhanced the naive MCTS approach and introduced a policy network guided MCTS method. This advanced approach leverages the strengths of MCTS and integrates it with the strategic direction provided by the policy network, aiming to optimize the generation process and yield high-quality ionizable lipids more efficiently.

1.2 Main Contributions

The main contributions of this work are as follows:

- **Construction of a lipid building block dataset** We compiled a comprehensive dataset of purchasable molecules suitable for use as ionizable lipid heads or tails. This building block dataset can facilitate future ionizable lipid generation tasks.
- **Development of a lipid classifier and an ionizability predictor** We develop lipid property predictors to assess whether a candidate molecule possesses lipid-like characteristics or ionizable properties. Both predictors demonstrate reliable performance.
- **Development of a policy network guided MCTS-based generative model for lipid generation** We propose and implement a generative model that leverages a policy network to guide the MCTS in exploring the chemical space. This model effectively generates high-quality products, complete with available synthesis pathways.

1.3 Outline

The main body of this dissertation is organised as follows:

- **Chapter 2** introduces the background of the project, including a chemical background of the fundamentals of ionizable lipids and a literature review of current machine learning research in molecular generation.
- **Chapter 3** details the construction process of our lipid building block dataset.
- **Chapter 4** covers all methodologies discussed in the dissertation, starting with the development of lipid property predictors. It then explores two MCTS-based generative approaches and includes the evaluation metrics used.

- **Chapter 5** presents key experimental results, discusses the effectiveness of our proposed methods in generating high-quality products, and examines the synthesis pathways identified by our model and their retrosynthesis performance.
- **Chapter 6** concludes the dissertation, discussing the limitations of the current project and outlining future work.

Chapter 2

Background

In this chapter, we provide a foundational overview essential for understanding our project, focusing on the fundamentals of ionizable lipids and the integration of machine learning in molecular generation. We begin by exploring the significance of ionizable lipids in pharmaceutical contexts, detailing their chemical and biological characteristics, as well as the synthesis challenges they present. We then shift our focus to the application of machine learning in molecular generation, starting with an introduction to molecular representations. This is followed by an examination of previous generative models used for molecular discovery and a discussion on the challenges associated with synthesizability in computer-aided molecular generations.

2.1 Fundamentals of Ionizable Lipids

Ionizable lipids are significant components of lipid nanoparticle (LNP) delivery systems. This section delves into the critical role and design of these lipids within LNPs. We begin with an introduction to LNPs, outlining their structure and function as delivery vehicles for nucleic acids. Subsequent discussions emphasize the importance of ionizable lipids, highlighting how their unique properties facilitate drug delivery. What's more, we also explore the evolution of ionizable lipid design, reviewing past researches that have shaped current strategies in generating ionizable lipids.

2.1.1 Introduction to Lipid Nanoparticles

Messenger RNA (mRNA) has become a pivotal tool in biomedicine, applied across a spectrum of fields including vaccine development, protein therapy, cell engineering, and gene editing (Hou et al., 2021; Qin et al., 2022). Its extensive applicability has generated substantial

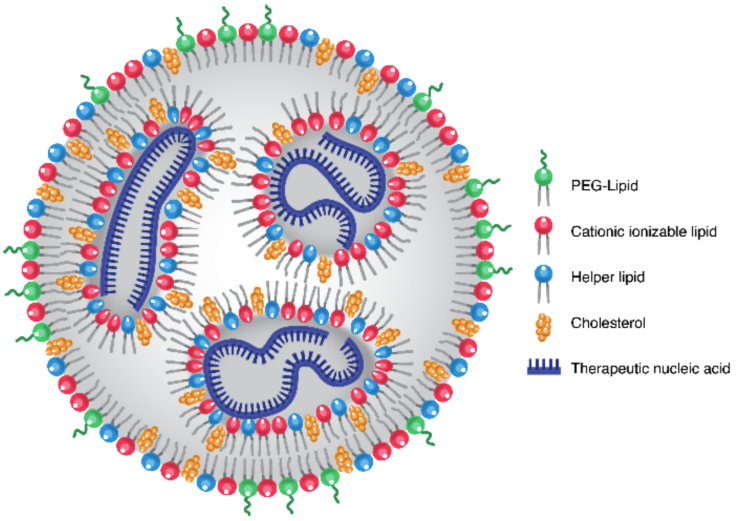


Fig. 2.1 LNP structure. (Figure sourced from (Kularatne et al., 2022) and adapted from (Evers et al., 2018))

interest in deploying mRNA to address diverse diseases (Kim, 2022; Mendes et al., 2022). However, the intrinsic instability of mRNA demands a robust delivery system, typically fulfilled by ionizable LNPs (Mitchell et al., 2021). LNPs as a drug delivery vehicle were first approved in 2018 for the siRNA drug Onpattro (Akinc et al., 2019; Rüger et al., 2020). LNPs became more widely known in late 2020 during the COVID-19 pandemic. The effectiveness of this delivery method was demonstrated by the success of Comirnaty and Spikevax, two SARS-CoV-2 vaccines approved amidst the COVID-19 pandemic, both of which employ LNP-mediated mRNA delivery. (Nasreen et al., 2022; Patrignani et al., 2021).

Currently, LNPs represent the cutting-edge technology for delivering therapeutic nucleic acids (Kularatne et al., 2022). The structure of LNPs is illustrated in Figure 2.1. LNPs have core-shell arrangement with distinct layers playing specific roles. A core central region is given by the cargo, such as mRNA in COVID-19 vaccines. After this, an inner shell layer surrounds the core and helps maintain its integrity. Finally, the outer shell layer directly interacts with the environment and plays a crucial role in the nanoparticle's behavior.

LNPs are composed of cholesterol, a helper lipid, a PEGylated lipid, and an ionizable amine-containing lipid (Eygeris et al., 2022). The cholesterol and helper lipids enhance the structural integrity and stability of the LNP, helping to maintain its shape and form (Kularatne et al., 2022). The PEGylated lipid helps maintain the stability of the particles and also gives them stealth qualities, which reduce their buildup in the body's cleansing system, known as the reticuloendothelial system (RES) (Kularatne et al., 2022). The ionizable amine-containing lipid is the most important. It facilitates the encapsulation of mRNA by

becoming positively charged in acidic conditions, and helps in releasing mRNA into the cell cytoplasm by disrupting endosomal membranes when re-protonated. Crucially, the ionizable lipid only gains a positive charge at a physiological pH, around 6–7 (Carrasco et al., 2021). This means it remains uncharged in the bloodstream, which is essential because positively charged nanoparticles can be highly toxic (Stern and McNeil, 2007).

2.1.2 Importance of Ionizable Lipids

As is mentioned before, the ionizable lipid is the most crucial component of the LNP formulation. Notably, each of the three FDA-approved RNA LNPs we discussed before features a unique ionizable lipid structure, emphasizing their crucial role in LNP technology. These lipids primarily help pack mRNA into LNPs and aid its delivery into the cell cytoplasm for protein production (Degors et al., 2019; Kim et al., 2021; Wittrup et al., 2015; Xu et al., 2021). An ionizable lipid typically has an ionizable amine head and two lipid tails (Xu et al., 2024). This design allows the lipid to become positively charged in acidic environments, aiding in the encapsulation of negatively charged RNA molecules during the LNP creation process. At physiological pH, these lipids are neutral, avoiding the toxicity seen with permanently charged lipids. When LNPs are taken into cells and reach the acidic environment inside endosomes, the ionizable lipids are protonated once more. This disrupts the endosomal membrane and releases the mRNA into the cytoplasm for its intended function (Han et al., 2021).

In the post COVID-19 era, the scope of mRNA applications continues to broaden beyond vaccination. The importance of developing a diverse range of ionizable lipids, capable of effectively delivering mRNA to a variety of target cells and tissues, thus becomes increasingly critical.

2.1.3 Ionizable Lipid Design

Over the past decade, high-throughput experimentation has transformed the pharmaceutical industry by enabling quick screening of compound libraries for therapeutic targets, it has also expanded towards the realm of small-molecule process chemistry (Mennen et al., 2019). Recently, combinatorial chemistry, with the aid of multi-component reactions, has facilitated the high-throughput synthesis (HTS) of extensive and chemically diverse lipid libraries (Xu et al., 2024). Using a three-dimensional multi-component reaction system, a combinatorial lipid library was generated and was then screened to identify a STING-activating ionizable lipid effective for mRNA vaccine delivery (Miao et al., 2019). Over 1 000 lipid formulations were synthesized and evaluated, revealing that the most effective ones

not only induced a strong immune response but also suppressed tumor growth and extended survival in melanoma and human papillomavirus E7 tumor models (Miao et al., 2019). More recently, using a high-throughput platform, a combinatorial library of biodegradable ionizable lipids was synthesized and screened to develop inhalable delivery vehicles for mRNA and CRISPR–Cas9 gene editors (Li et al., 2023). The lead lipid nanoparticles, suitable for repeated intratracheal dosing, achieved efficient gene editing in lung epithelium, offering potential for gene therapy in congenital lung diseases (Li et al., 2023).

Although HTS has been showcased to facilitate the synthesis of ionizable lipids, constructing and testing a more extensive lipid library for other biomedical properties like mRNA transfection in different cell targets remains a time-consuming and costly task (Han et al., 2021; Xu et al., 2024). AI-Guided Ionizable Lipid Engineering (AGILE) platform pioneered combining deep learning methodologies with combinatorial chemistry to accelerate the development of ionizable lipids for mRNA delivery (Xu et al., 2024). Specifically, AGILE first developed a graph encoder to differentiate and depict distinct lipids through pre-training on a vast unlabeled lipid dataset. The model was further trained with mRNA transfection potency data derived from a selection of ionizable lipids. The trained model was then used for in silico screening to predict and screen for the most effective lipid-based carriers for mRNA delivery (Xu et al., 2024). The success of AGILE demonstrates a promising future for using deep learning to rapidly design, synthesize, and evaluate new ionizable lipids for mRNA delivery.

2.2 Machine Learning in Molecular Generation

Machine learning has proven to be a powerful tool for molecular generation. In this section, we first explain how molecules can be represented for computational processing. We then delve into the latest generative models that are shaping the field of molecular generation. Following this, we address the critical challenge of synthesizability, discussing recent advancements aimed at overcoming this hurdle.

2.2.1 Molecular Representation

In the realm of machine learning for molecular generation, the representation of molecules plays a pivotal role in determining the effectiveness of predictive models and generative systems. Molecular representation refers to the method of encoding molecular structures into a format that can be processed by machine learning algorithms.

Simplified Molecular Input Line Entry System (SMILES)

SMILES is a widely used notation that encodes the structure of a molecule as a line of text (Weininger, 1988). This representation captures the arrangement of atoms and bonds within a molecule in a compact form, making it suitable for input into various machine learning models. However, while SMILES is efficient for data storage and transfer, it does not directly capture 3D molecular geometry.

Molecular Graphs

Molecular graphs represent molecules as graphs where atoms are nodes and bonds are edges. This representation naturally accommodates the relational information between atoms and can be directly utilized in graph-based machine learning models such as Graph Neural Networks (GNNs) (Bronstein et al., 2021). These models have shown significant success in predicting molecular properties and activities by exploiting the inherent graph structure of molecular data (Kipf and Welling, 2016; Veličković et al., 2018).

Fingerprint Vectors

Fingerprint vectors are binary vectors that offer a fixed-size representation of molecules, facilitating the systematic exploration of molecular structures. These vectors are produced through algorithms that identify the presence or absence of specific substructures or molecular features within a molecule (Schneider et al., 2015). Additionally, developments in count-based fingerprints have enhanced this model by quantifying the occurrences of certain substructures within a molecule (Zhong and Guan, 2023). Fingerprint vectors are particularly useful in similarity searches and classification tasks.

2.2.2 Generative Models for Molecular Discovery

Deep generative models have emerged as a powerful solution to the inverse molecular design challenge, enabling the translation of desired molecular properties into specific molecular structures. These models are designed to quickly produce a wide array of molecules tailored for particular uses. In the last few years, there has been a surge in various generative model architectures for molecular generation, like variational autoencoders (VAEs) (Gómez-Bombarelli et al., 2018; Kusner et al., 2017; Liu et al., 2019; Simonovsky and Komodakis, 2018), generative adversarial networks (GANs) (Assouel et al., 2018; Bian et al., 2019), normalizing flow models (Madhawa et al., 2019; Zang and Wang, 2020), diffusion models

(Luo et al., 2021; Shi et al., 2021; Xu et al., 2022) and reinforcement learning methods (Simm et al., 2020; You et al., 2018).

However, a practical problem that obstructs the usefulness of these generative algorithms is that proposed molecular structures may be challenging or infeasible to synthesize (Gao and Coley, 2020). Without feasible methods for the chemical synthesis of these in silico-generated molecules, their experimental validation and functional testing become impractical. Therefore, designing generative models that take synthesizability into consideration is of great significance.

2.2.3 Synthesis Planning in Molecular Generation

Synthesis planning is the process of determining how to synthesize a chemical compound from available starting materials through a series of chemically feasible reaction steps (Coley et al., 2018). While post-hoc synthesis planning for generated molecules is feasible (Gao and Coley, 2020; Segler et al., 2017, 2018), a more effective approach is to incorporate synthesis instructions directly into the design phase (Bradshaw et al., 2020).

To effectively integrate synthesis planning into molecular generation, adopting a bottom-up approach can be highly advantageous. This method begins with existing building blocks and strategically determines pathways to synthesize product molecules possessing desired properties. The DOG framework exemplifies this innovative approach by concurrently developing synthetic pathways alongside new molecules (Bradshaw et al., 2020). It utilizes a novel neural architecture that represents multi-step molecular synthesis routes as directed acyclic graphs (DAGs). This allows DOG not only to generate novel molecules but also to directly produce the corresponding synthesis DAGs, ensuring that the design process is intimately linked with practical synthesizability. While DOG was initially designed to generate small drug-like molecules, its direct applicability to lipids—which are typically larger and possess more complex structures—may be limited.

Instead of relying on generative models to generate synthesis routes, another viable approach involves direct exploration within the combinatorial chemical space. SyntheMol provides a compelling example of this strategy, utilizing Monte Carlo Tree Search (MCTS) to effectively navigate through a vast chemical space (Swanson et al., 2024). This approach not only generates desired molecules but also concurrently develops their synthesis pathways, particularly demonstrated in the context of antibiotic development.

2.3 Monte Carlo Tree Search

In this section, we provide the theoretical background and review related literature concerning our Monte Carlo tree search (MCTS)-based generative methods. We begin by introducing SyntheMol, an MCTS-based generative model developed for antibiotic discovery. This model sets the groundwork for understanding how MCTS can be effectively utilized in molecular generation. Following this, we delve into the enhancements of MCTS made through the integration of policy networks, detailing how these adaptations improve the model's ability to navigate the search space.

2.3.1 Monte Carlo Tree Search in Molecular Generation

MCTS is a robust algorithm that integrates the stochastic nature of Monte Carlo simulations with the structured decision-making processes inherent in tree searches (Coulom, 2006; Kocsis and Szepesvari, 2006). At the core of MCTS is the iterative construction of a decision tree, where simulations evaluate potential moves based on their likelihood of leading to a successful outcome. This method, merging the depth of tree search with the breadth of random sampling, adeptly manages the exploration-exploitation tradeoff. Such capability is crucial in fields requiring sophisticated strategy and foresight, particularly making it highly effective for molecular generation in vast combinatorial chemical spaces.

Recent study has shown that MCTS based generative models can perform successfully in small-molecule antibiotic development (Swanson et al., 2024). SyntheMol leverages a MCTS strategy to design new, synthesizable compounds (Swanson et al., 2024). MCTS is a decision-making algorithm that explores potential moves in a vast space by building a tree of decisions. In the context of SyntheMol, this involves assembling novel compounds using about 132,000 molecular building blocks with known reactivities. The algorithm iteratively builds a tree structure, where each node represents one or more potential molecular structures, and branches represent possible synthesis steps using 13 well-validated chemical reactions.

The MCTS process prioritizes the exploration of chemical spaces that have a higher probability of yielding synthesizable and effective molecules. It balances between exploring new areas of the chemical space and exploiting known areas to refine promising candidates. This ensures that the molecules designed are not only structurally novel but also have a practical synthesis pathway, addressing a common issue with generative models that often propose molecules that are challenging to synthesize. While most generative AI methods designed for synthesizable molecules have been proposed with promising in silico results (Bradshaw et al., 2019, 2020; Gao et al., 2021; Gottipati et al., 2020; Pedawi et al., 2022), researchers of the SyntheMol study synthesized and experimentally validate their

generated molecules. With a synthesis success rate of over 80% within 3–4 weeks, SyntheMol demonstrates the effective use of MCTS in generating viable and active antibiotic compounds.

2.3.2 Guided Monte Carlo Tree Search

The pioneering work that integrates neural networks with MCTS is AlphaGo (Silver et al., a), which marked a significant milestone in the application of AI to strategic games like Go. AlphaGo utilized two neural networks: a policy network that guided the search by predicting the next best move, and a value network that evaluated potential game outcomes from current positions. This dual-network setup enabled a profound and effective exploration of possible moves, substantially enhancing strategic positioning and predictive capabilities.

Building upon the foundation set by AlphaGo, AlphaZero further refined this approach by merging the policy and value networks into a single, more efficient neural network (Silver et al., b). Unlike AlphaGo, which initially trained its policy network using supervised learning from human expert games, AlphaZero was trained exclusively through self-play. This training method allowed AlphaZero to autonomously develop and refine strategies, enabling the system to adapt and optimize its game play without relying on external data, representing a leap forward in AI autonomy.

When considering the application to molecular generation, the AlphaZero algorithm provides a highly relevant framework for our method. Although our approach diverges from AlphaZero in that it does not integrate a value network within the MCTS framework—instead relying on a separate property predictor—the role of the policy network remains pivotal in both contexts. In Go, AlphaZero's policy network determines the optimal position for stone placement, thereby optimizing game strategies by predicting opponents' responses and the consequential long-term impacts. Similarly, in molecular generation, our policy network is instrumental in selecting the next building block for synthesis pathways. This selection directly influences the structure and properties of the resultant molecules, akin to how each move in Go influences the progression and outcome of the game.

This analogy underscores how strategic decision-making, facilitated by AlphaZero's policy network, can be effectively adapted to the complex challenges of molecular generation. By directing the MCTS to explore the most promising synthetic routes, the policy network efficiently navigates vast chemical spaces, focusing on pathways most likely to yield successful synthesis and optimal molecular properties.

Chapter 3

Dataset Construction

This chapter presents the detailed lipid building block dataset construction process. Our method applies a bottom-up strategy to generate ionizable lipids. We first show the whole process of constructing ionizable lipid head dataset and lipid tail dataset, and then separately display our constructed building block datasets in detail. Note that the dataset creation process is a collaborative effort with Yuxuan Ou.

3.1 Method

An ionizable lipid consists of an ionizable lipid head and several lipid tails. In order to find the synthesizable pathway to generate ionizable lipids, we adopt a bottom-up approach which selects lipid heads and lipid tails from building block datasets. The building block datasets contain purchasable, i.e., publicly accessible, molecules that can act as lipid heads or lipid tails. We filter valid lipid building blocks from the ZINC20 database (Irwin et al., 2020), a large-scale chemical database designed for drug discovery and known for providing publicly accessible chemical compounds for virtual screening.

Figure 3.1 shows the flowcharts for constructing lipid head and lipid tail building block datasets. For the ionizable lipid head dataset, we first filter ionizable molecules from the ZINC20 molecule dataset based on the following three criteria:

1. Molecular weight < 500 g/mol.
2. Log P < 0 where the log P is the logarithm (base 10) of the partition coefficient P, which is the ratio of the concentrations of a compound in a mixture of two immiscible phases: typically a hydrophobic solvent and water.
3. Molecules with amine functional groups but not ammonium based molecules.

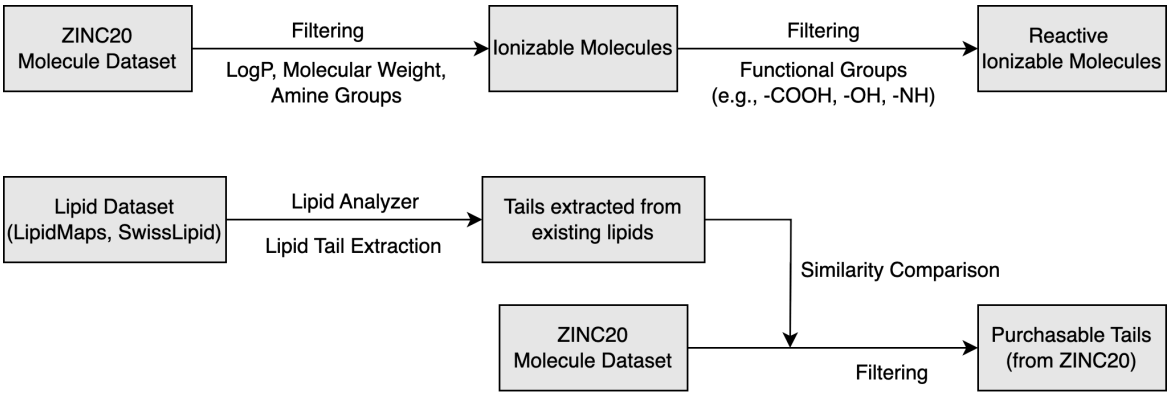


Fig. 3.1 Roadmap of lipid building block datasets construction. The upper flowchart illustrates the process of filtering ionizable lipid head building block dataset. The lower flowchart illustrates the process of constructing lipid tail building block dataset. Both datasets come from publicly accessible molecule dataset ZINC20.

Since the lipid head building blocks are expected to react with lipid tail building blocks to generate product lipids, we further filter reactive molecules from our ionizable molecule set. We identify reactive molecules by filtering whether or not the molecule contains certain functional groups that participate in common reactions. Specifically, we check whether our candidate molecule contains carboxyl group (i.e., -COOH), hydroxyl group (i.e., -OH), or amine group (i.e., -N or -NH or -NH2). If the candidate molecule contains any one or more of the target functional groups, we consider the molecule to be reactive. Our filtered reactive ionizable molecule set becomes our ionizable lipid head dataset.

In terms of lipid tail building block dataset, we again filter from the ZINC20 molecule dataset so that our lipid tail building blocks are purchasable. We identify a lipid tail building block if the molecule is similar to an actual lipid tail (i.e., the tail component of an existing lipid). We use a lipid dataset sourced from the LipidMaps database and the SwissLipid database (Aimo et al., 2015; Sud et al., 2007). Taking advantage of the Lipid Analyzer[1], an implemented toolkit for lipid tail extraction, we extract lipid tails from this lipid dataset. The Lipid Analyzer finds the lipid head for a given lipid, we then extract lipid tails by removing the head substructure. For a given molecular structure, the algorithm first recognizes ring structures. For all possible arrangements of the rings, the algorithm then removes carbon atoms that are not near any hydrophilic atom and not forming any ring. If only one substructure remains after the operation, and if the log P value of this substructure is low enough, this substructure will be identified as the lipid head. It's then easy to extract tail substructures by removing the head substructure.

[1]The Lipid Analyzer toolkit is developed by Mohammad Shahab Sepehri from University of Southern California, the source codes are not publicly available.

We then conduct a similarity comparison to filter those molecules from the ZINC20 dataset that are similar to a real lipid tail. The similarity is measured by the Graph Edit Distance (GED) between two molecular structures, quantifying the minimum number of operations required to transform one graph into another. We only select molecules that has GED value smaller or equal to 1 with a real lipid tail. Meanwhile, we also consider the similarity between molecular fingerprint representations, Daylight fingerprint and Extended Connectivity Fingerprint 4 (ECFP4) are considered. The resulting molecules make up our lipid tail building block dataset.

3.2 Lipid Head Dataset

The ZINC20 dataset has more than 1 billion molecules in total. Following the lipid head building block dataset construction process described above, we get more than 33 million molecules after filtering via molecular weight and log P values. We further refine the candidate molecule set to 16.4 million after filtering via amine functional groups. Note that one molecule may contain several target functional groups (i.e., -COOH, -OH, or -N), we then filter reactive molecules based on how many functional groups each molecule contains.

Table 3.1 Number of Lipid Head Molecules with Different Functional Groups

No. Carboxyl	**No. Hydroxyl**	**No. Amine**	**Number of Molecules**
0	0	2	76784
0	1	1	1454956
0	1	2	204324
0	1	3	17039
0	2	1	164071
0	3	1	12105
1	0	1	73175
1	0	2	26322
1	0	3	4496
1	1	1	19870
1	1	2	4883
1	2	1	2758
2	0	1	516
2	1	1	105
3	0	1	16

Table 3.1 shows the number of molecules we get based on different combinations of the number of target functional groups. Our ionizable lipid head building block dataset has above 2.7 million molecules in total.

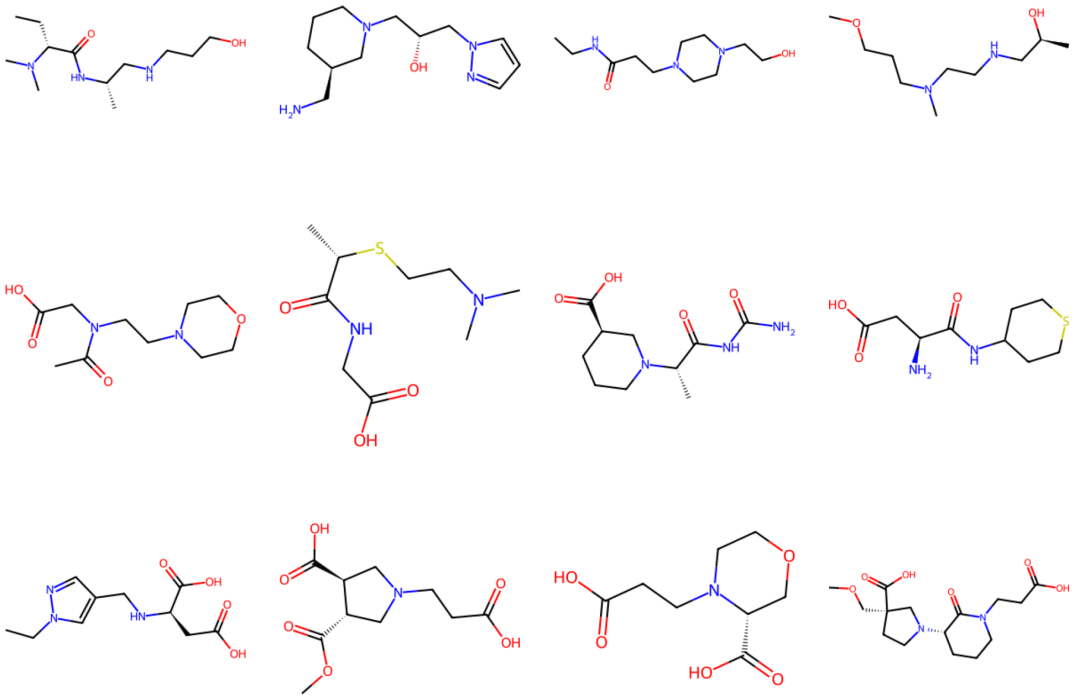


Fig. 3.2 Selected examples of lipid head building blocks with different functional groups.

Selected examples of our lipid head building blocks with different functional groups are shown in Figure 3.2. The first row presents examples with no carboxyl group, one hydroxyl group, and two amine groups; the second row presents examples with one carboxyl group, no hydroxyl group, and one amine group; the third row presents examples with two carboxyl groups, no hydroxyl group, and one amine group. Note that we only consider independent amine groups that are linked to only simple carbon atoms. For example, in the first example of the first row, we find a secondary amine group (i.e., -NH) linked to a carbonyl group (i.e., C=O, a carbon atom double-bonded to an oxygen atom). The linkage with the carbonyl group forms a more complicated substructure, which influences the reactive performance of the amine group. We therefore exclude this secondary amine group when counting our target functional groups. Similarly, the nitrogen-nitrogen bond (i.e., N-N) which appears in a five-membered nitrogen-containing ring influences the reactive property of the nitrogen, and we also exclude these nitrogen atoms when counting the occurrence of amine functional groups.

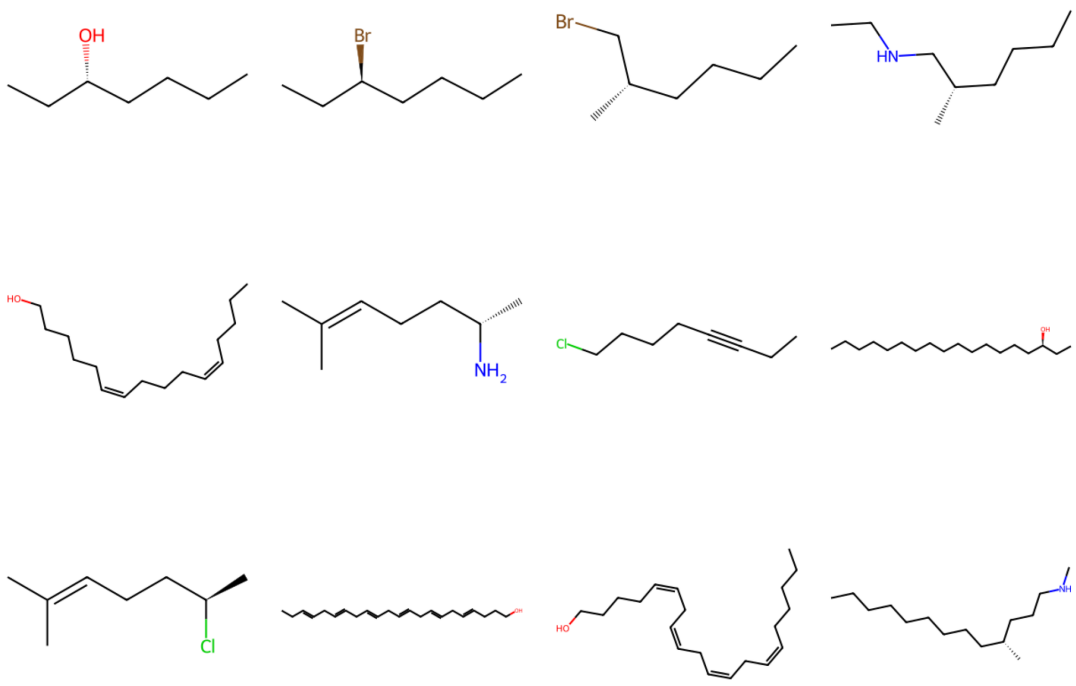


Fig. 3.3 Selected examples of lipid tail building blocks.

3.3 Lipid Tail Dataset

We apply Lipid Analyzer to around 60 000 lipid molecules from LipidMaps, resulting in 5423 lipid tails with heteroatoms (i.e., atoms other than carbon or hydrogen) and 2753 lipid tails without heteroatoms. We only focus on lipid tails without heteroatoms in our experiment. After the similarity comparison with ZINC20 molecules, we construct a lipid tail building block dataset with 5310 unique purchasable molecules.

Figure 3.3 shows selected examples of lipid tail building blocks. As we can see, a lipid tail usually consists of a carbon chain and a functional group. The functional group may be hydroxyl group, amine group, or a halogen atom.

Chapter 4

Methodology

This chapter presents the Monte Carlo Tree Search (MCTS) based generative models for generating ionizable lipids from lipid building block dataset. We first present the lipid property predictors, lipid classifier and ionizability predictor, for evaluating whether the generated product has desired properties. The property predictors play important roles in guiding the MCTS. We then present two MCTS based approaches. The naive MCTS approach adopts a simple Upper Confidence Bound (UCB) action selection criteria to explore the combinatorial chemical space. This approach serves as our baseline model. The policy network guided MCTS approach takes advantage of a neural network to decide the action selection.

4.1 Lipid Property Prediction

Our generative approach relies on a molecular property prediction model to evaluate the potential of generated molecules to be an ionizable lipid. We use a binary classification model, lipid classifier, to determine whether or not the generated product is lipid-like. In terms of ionizability, although we've filtered ionizable molecules when constructing lipid head building block dataset, the ionizability may be broken after reactions with tail building blocks. It's therefore crucial to establish an ionizability predictor to determine whether or not the generated product is ionizable.

4.1.1 Lipid Classifier

The lipid classifier is expected to give a binary output indicating whether or not the given molecule is lipid-like. We build the lipid classifier using a graph neural network. The dataset we use to train the lipid classifier contains 180 000 lipid samples and 180 thousand non-lipid

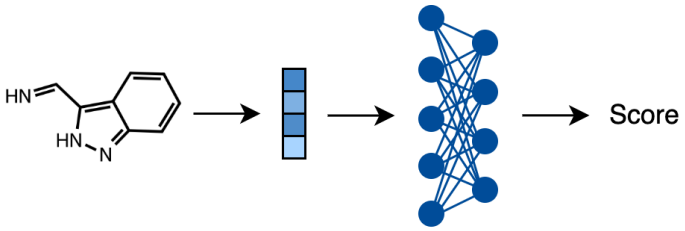


Fig. 4.1 Simplified visual depiction of the Chemprop property prediction model.

samples. The non-lipid samples are small molecules from the PubChem database (Kim et al., 2016). Part of the lipid samples come from publicly accessible lipid dataset LipidMaps and SwissLipids (Aimo et al., 2015; Sud et al., 2007). The rest of the lipid samples are generated using a hierarchical graph encoder-decoder that employs significantly large and flexible graph motifs as basic building blocks (Jin et al., 2020).

We train a Chemprop model to be our lipid classifier (Yang et al., 2019), a simplified visual depiction of the model is shown in Figure 4.1. Chemprop is a molecular property prediction model that employs a graph neural network for processing molecules and predicting their properties. It begins by extracting basic features from the molecular graph, including the type of each atom and bond. These features are used to construct feature vectors for each atom and bond. Chemprop then carries out three rounds of message passing, using a neural network layer to progressively integrate information from adjacent atoms and bonds. After these message passing steps, it aggregates the combined feature vectors into a single vector that encapsulates the entire molecule. This comprehensive feature vector is subsequently processed by a two-layer feed-forward neural network to predict the molecular property.

4.1.2 Ionizability Prediction

What we need from an ionizable molecule is to have near zero charge in physiological pH (i.e., pH = 7-7.5) and become positively charged in acidic environments (i.e., pH < 5). The process of calculating the net charge of a molecule under a given pH is described below. For simpler filtering purpose, we only consider the net charge under pH = 7.4 (i.e., represents the physiological pH) and pH = 5 (i.e., represents the acidic environment). The roadmap of our ionizability filtering process is depicted is Figure 4.2.

For a given molecule, we first identify all the ionizable groups in the molecule and their respective pKa values. We make use of the MolGpKa module to determine acidic and basic groups within a molecule and to estimate their respective pKa values (Pan et al., 2021). Extending the SMARTS representation list of 38 ionizable substructures provided by (Ropp et al., 2019), MolGpKa proposed a SMARTS list containing 144 ionizable groups so that

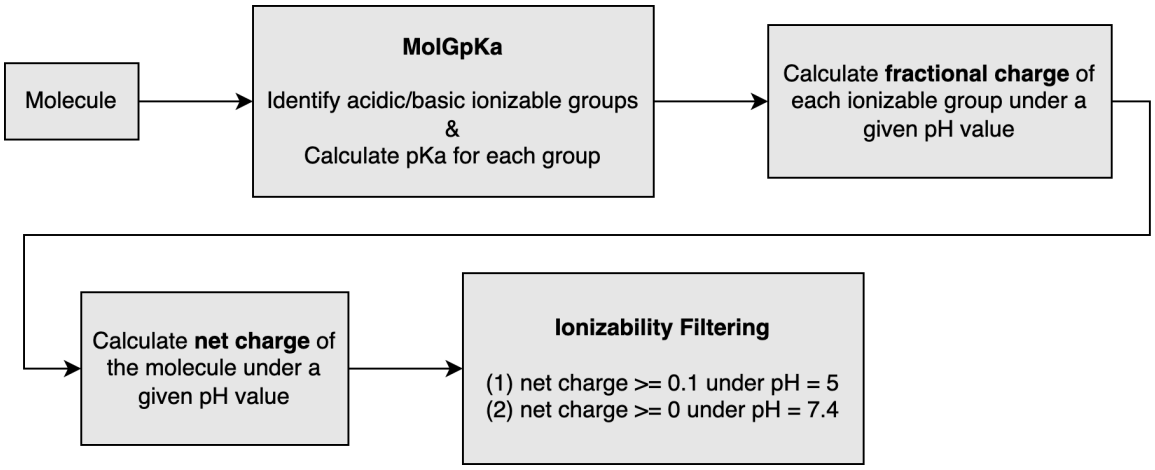


Fig. 4.2 Roadmap of ionizability filtering via charge-pH.

all ionizable oxygen, nitrogen, and sulfur centers are covered. Given the SMARTS list, all ionization centers are identified by substructure matching with RDKit (RDKit, nd). MolGpKa trains two separate models to predict pKa values for acidic and basic ionizable groups using graph convolutional neural networks (Kipf and Welling, 2016). The neural network learns important chemical features and physical principles that are related to the acidity (or basicity) of the root atom and makes reliable pKa predictions. The model was trained using computational pKa values of bioactive compounds in the ChEMBL database(Gaulton et al., 2012).

We then calculate the fractional charge of each ionizable group under a given pH value using the Henderson-Hasselbalch equation (Petrucci et al., 2002):

$$pH = pKa + \log \frac{[Base]}{[Acid]} \tag{4.1}$$

where the [ ] notation means concentration. When applying to the weak acid scenario $\underset{\text{(acid)}}{\text{HA}} \leftrightharpoons \underset{\text{(base)}}{\text{A}^-} + \text{H}^+$, we get the following equation:

$$pH = pKa + \log \frac{[A^-]}{[HA]} \tag{4.2}$$

Similarly, when applying to the weak base scenario $\underset{\text{(base)}}{\text{B}} + \text{H}^+ \leftrightharpoons \underset{(acid)}{BH^+}$, we get the following equation:

$$pH = pKa + \log \frac{[B]}{[BH^+]} \tag{4.3}$$

The fractional charge for acids can be calculated as

$$\text{fractional charge} = \frac{10^{pH-pKa}}{1+10^{pH-pKa}} \tag{4.4}$$

And the fractional charge for bases can be calculated as

$$\text{fractional charge} = \frac{10^{pKa-pH}}{1+10^{pKa-pH}} \tag{4.5}$$

Note that the fractional charge values in the above equations are absolute values. In reality, acids typically get negative charges while bases get positive charges.

Given the fractional charges of each acidic group and each basic group, we can calculate the net charge of a molecule by summing the fractional charges of all ionizable groups within the given molecule.

Recall that an ionizable molecule is expected to have neutral charge at physiological pH and positive charge in acidic environments. We therefore set our first filtering criterion to be net charge >= 0.1 under pH = 5. In terms of physiological pH, in practice, we relax the neutral charge condition to neutral or positive charge. The reason is that our ionizable lipid will ultimately participate in the formation of a LNP, which contains four types of lipids (i.e., our ionizable lipid will be one of the four types), and the LNP is expected to be neutrally charged under physiological pH. It's therefore possible for our ionizable lipid to be positively charged under physiological pH while maintaining the whole LNP to be neutrally charged. Our second filtering criterion is thus set to be net charge >= 0 under pH = 7.4. If the given molecule passes the two filtering criteria, the molecule is predicted to be ionizable.

In summary, we take advantage of the MolGpKa module to identify ionizable groups of a given molecule and predict pKa values for each ionizable group. The net charge can then be calculated based on the pKa values of these ionizable groups when specifying a pH value. We predict the molecule to be ionizable if the net charge >= 0.1 under pH = 5 and if the net charge >= 0 under pH = 7.4.

4.2 Reaction Prediction

Recent advancements in chemical informatics have seen the development of machine learning based forward reaction prediction models that predict the outcomes of chemical reactions using historical data (Irwin et al., 2022; Schwaller et al., 2019). These models employ various machine learning techniques to learn from vast datasets of chemical reactions, enabling them to propose potential reaction products given specific reactants.

In contrast to machine-learning models, our project employs a template-based reaction prediction model, which offers several advantages particularly aligned with our objectives of synthesizing ionizable lipids. Template-based models operate by applying predefined reaction templates—derived from known chemical reaction mechanisms—to the reactants in question. The primary benefits of using a template-based reaction prediction model include faster computation, strict adherence to established chemical rules, and enhanced potential for successful synthesis. Unlike machine learning models that require intensive processing for on-the-fly predictions, template-based models utilize a fixed set of reaction rules, allowing for rapid generation of potential products with reduced computational overhead. These models strictly follow recognized chemical transformation rules, ensuring that the synthetic routes proposed are feasible and effective in practical laboratory settings, which is essential for synthesizing ionizable lipids. By focusing on reactions known to be successful, template-based models enhance the likelihood that the predicted products can be reliably synthesized under typical laboratory conditions.

We adopt the same reaction templates that are used by SyntheMol (Swanson et al., 2024), which acts as our baseline model. The combinatorial chemical space used by SyntheMol was the Enamine REadily AccessibLe (REAL) Space which consists of 31 billion molecules producible via 169 chemical reactions from 138 thousand molecular building blocks (Grygorenko et al., 2020). Specifically, SyntheMol employs 13 reactions that account for 93.9% of the REAL Space (Swanson et al., 2024). While our lipid building block dataset differs from the REAL space, the selected reactions are widely applicable across a broad range of chemicals, making them suitable for our project as well. A detailed list of these reaction templates is provided in Appendix B.

4.3 Naive Monte Carlo Tree Search for Lipid Generation

Our objective is to generate ionizable lipids that are synthesizable. Now that we have the lipid building block dataset and a template-based reaction prediction model, generating the synthesis path for a product molecule with desired properties essentially becomes a combinatorial problem. We apply a Monte Carlo tree search (MCTS) guided by a property prediction model to search through the vast combinatorial chemical space for promising ionizable lipid candidates. Specifically, the state-action space is the chemical space constructed by the provided lipid building block datasets and reaction templates.

Leveraging the success of the SyntheMol approach in the generation of small-molecule antibiotics using MCTS, we can adapt this strategy for the generation of ionizable lipids. The SyntheMol methodology, which effectively combines vast chemical space exploration with a

guided synthesis approach (Swanson et al., 2024), provides a solid foundation for our lipid generation project. Specifically, the adaptability of this method to incorporate a variety of molecular building blocks makes it highly suitable for synthesizing ionizable lipids, where the selection and combination of lipid head and tail molecules are critical. We therefore adopt the SyntheMol method as our baseline model, which we refer to as the naive MCTS approach. This serves as a point of comparison for the guided MCTS approach, discussed in the subsequent section, which incorporates neural networks.

Building on the SyntheMol framework, our lipid generation process begins with the curation of a large lipid building block dataset, as discussed in Chapter 3. To simulate feasible synthetic pathways, we employ a template-based reaction prediction model, detailed in Section 4.2. This model predicts forward reactions by mapping known reaction templates to the available building blocks, effectively generating potential product molecules. The combination of the building block dataset and the reaction templates defines the vast chemical space to be explored. In parallel, as previously discussed in Section 4.1, a property prediction model plays a crucial role in evaluating the potential of these synthesized products to function as ionizable lipids.

With the chemical space and predictive models in place, we utilize an MCTS-based algorithm to efficiently explore this space. Figure 4.3 illustrates a typical simulation of the MCTS process in our lipid generation project. In the search tree, each node stores crucial statistics: the molecules represented by the node, the node's visit count N, and the Upper Confidence Bound (UCB) score. The UCB score is critical for navigating the search tree as it guides the node selection process using the UCB action selection criterion (Kocsis and Szepesvari, 2006; Sutton and Barto, 2018). Specifically, nodes with the highest UCB scores within their level are selected during the search process.

The UCB score combines an action value Q, which represents the average of the total action values W (i.e., the sum of the values of the final products passing through the node), with an upper confidence term U. This term is defined as $U(node) \propto \frac{P(node)}{1+N(node)}$, where $P(node)$ indicates the property scores assigned to the node. The complete formula for the UCB score is as follows:

$$UCB\_score(node) = Q(node) + U(node) \tag{4.6}$$

$$= \frac{W(node)}{N(node)} + c \cdot P(node) \cdot \frac{\sqrt{1+n}}{1+N(node)} \tag{4.7}$$

where n is the total visit count of all nodes at the same level as the target node and c is a constant exploration parameter controlling the level of exploration. This scoring mechanism balances exploitation of known pathways with the exploration of less frequented paths,

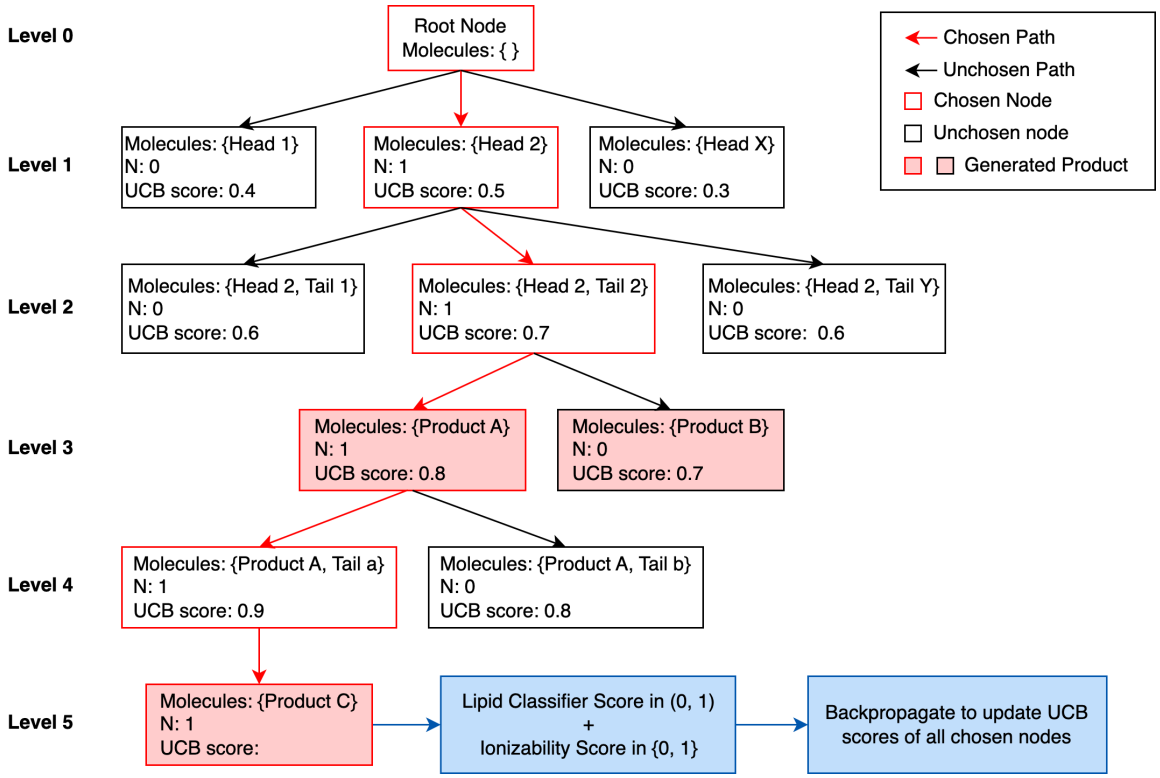


Fig. 4.3 Naive Monte Carlo Tree Search for lipid generation.

effectively guiding our search through the complex chemical space. Note that the P value is the property score calculated by our property predictors. When a node represents multiple molecules, the P value for that node is computed as the average of the property scores for each constituent molecule. Notably, a major deviation from the standard MCTS workflow in our approach is that we do not conduct rollouts for node value estimation. Instead, we directly compute the UCB values as previously described.

It is important to note that the assignment of property scores to nodes at non-terminal levels—representing either building block molecules or intermediates—may initially seem illogical, as direct evaluation of these entities' properties does not typically yield meaningful insights. However, this approach does not compromise the long-term efficacy of the algorithm. Over time, as more product molecules are synthesized and action values refined, the utility of early-stage evaluations is validated. SyntheMol's findings corroborate this approach, demonstrating that despite low scores of individual building blocks, the synthesized molecules often exhibit significantly higher scores, which are effectively identified by MCTS, highlighting its capability to uncover promising molecules overlooked by simpler scoring methods (Swanson et al., 2024).

The MCTS algorithm commences at a root node (level 0), an empty initial node, which is then expanded with child nodes representing available lipid head building blocks defined within our chemical space (level 1). Each child node receives an UCB score, with the node exhibiting the highest UCB score selected for further expansion. This selected node is expanded to include a lipid tail building block, generating second-level nodes where each combination represents potential reactants (level 2).

Subsequently, the node with the highest UCB score from level 2 is selected and expanded. This stage differs from prior expansions in that it now entails actual chemical reactions between the two building blocks within the node. The resulting third-level nodes embody all conceivable products predicted by forward reaction mechanisms (level 3). This iterative expansion continues until either a valid final product (i.e., a two-tail lipid in our case) is synthesized, or further reactions become untenable with the existing building blocks in our dataset.

Upon terminating the simulation with a valid final product, the synthesized molecule is assessed using our property predictors. The resultant property value is then backpropagated to update the UCB scores along the simulated pathway. For analysis purpose, we meticulously document all products, including intermediates, generated during each simulation.

4.4 Guided Monte Carlo Tree Search for Lipid Generation

Building on the foundation laid by the naive MCTS approach, which utilizes the tree search with a simple UCB action selection criterion, we enhance the model's capabilities by incorporating a policy network into the MCTS framework. This advancement, termed policy network guided MCTS, integrates deep learning to refine the selection process within the tree search, guiding it towards more promising molecular structures. Specifically, the major difference with the naive MCTS is to replace the P values in the previous action selection equation (i.e., Equation 4.7) with a neural network output. This section delves into the nuances of how the policy network guided MCTS can be applied to lipid generation,

4.4.1 Method Overview

Before delving into the specific steps of our policy network training procedure, it is essential to establish a clear understanding of the framework within which this process operates. The integration of a policy network with MCTS forms the cornerstone of our approach, enabling a strategic exploration of chemical spaces through guided decision-making. The training of the policy network is a cyclic process, aimed at incrementally enhancing the network's ability

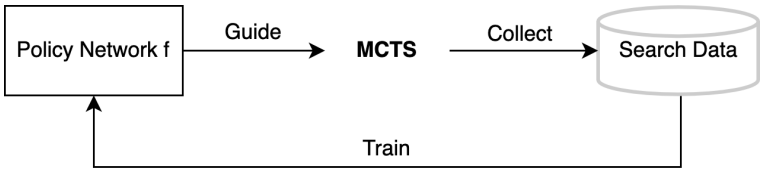


Fig. 4.4 Workflow of policy network training.

to predict and prioritize effective synthetic pathways. This ongoing refinement is crucial for optimizing the molecular generation process, ensuring that each iteration contributes to a more sophisticated and capable system. The workflow for this training procedure, as detailed in the following paragraph, outlines the iterative nature of the process, highlighting how the network evolves through successive cycles of training and application.

Figure 4.4 illustrates the workflow of the policy network training procedure . We start from a randomly initialized policy network which assigns equal probabilities to all the actions in the provided action space. This policy network will be used to guide the MCTS. We conduct the tree search for a number of simulations, and the search data (i.e., visit counts of all state-action pairs involved) of the tree search will be used as the training data for the policy network. The policy network will be trained for several epochs using these training data. This process serves as one iteration of policy network training. In the next iteration, we use the updated policy network to guide MCTS and repeat the process.

The search data we get from the MCTS give the search probabilities of choosing each action at a given state. These search probabilities usually select much stronger actions than the raw move probabilities of the policy network, and thus MCTS may be viewed as a powerful policy improvement operator (Howard, 1960; Silver et al., b; Sutton and Barto, 2018). The MCTS-based generation, using the improved policy network to select the next building block molecule and evaluate using property predictors, may be viewed as a powerful policy evaluation operator (Silver et al., b). The main idea of our reinforcement learning algorithm can thus be viewed as a policy iteration procedure: the policy network's parameters are updated to make the move probabilities more closely match the improved search probabilities; these updated parameters are used in the next iteration to make the search even stronger.

A detailed depiction of steps involved in each simulation of the MCTS is shown in Figure 4.5. Each simulation of the MCTS consists of four steps: select, expand, rollout, and backpropagate. We define the policy network that guides the MCTS simulations to be f_θ with parameters θ. Each edge in the tree search represents a state-action pair (s,a) where state s is the current molecule we have and action a is the next building block molecule to choose. Each edge stores a set of statistics $\{N(s,a),W(s,a),P(s,a)\}$ where $N(s,a)$ is the

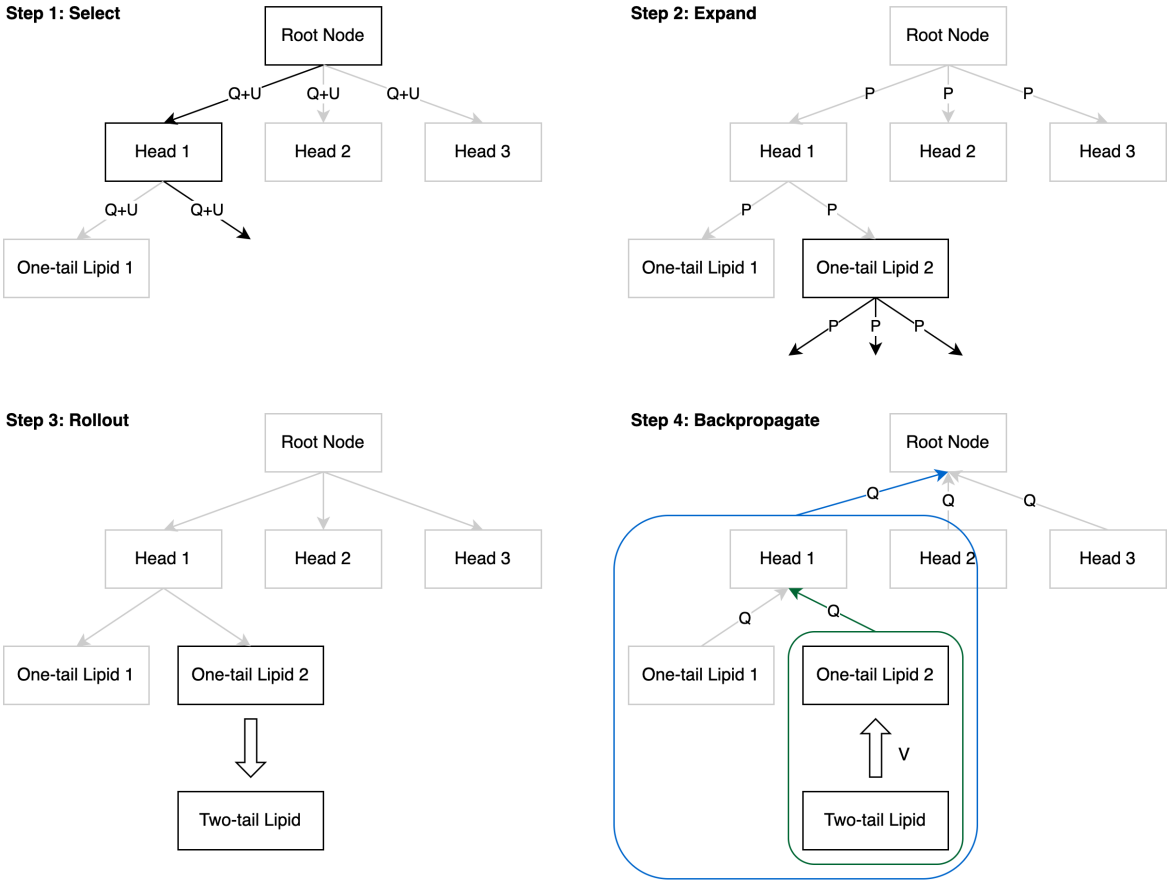


Fig. 4.5 Policy network guided Monte Carlo Tree Search for lipid generation.

visit count, $W(s,a)$ is the total action value (i.e., sum of values of final products reached after taking a from state s), and $P(s,a)$ is the prior probability of selecting that edge. Note that $P(s,\cdot) = f_\theta(s)$ is given by the current policy network.

In the selection step, each simulation traverses the tree by selecting the edge with the maximum UCB score until a leaf node is reached. The UCB score is defined to be an average action value $Q(s,a)$ plus an upper confidence bound $U(s,a)$ which depends on the prior probability $P(s,a)$ and the visit count $N(s,a)$ of that edge. At each time step t of each simulation, our action selection criterion at state s_t follows

$$a_t = \arg\max_a UCB\_score(s_t,a) = \arg\max_a (Q(s_t,a) + U(s_t,a)) \tag{4.8}$$

The UCB score is defined to be

$$UCB\_score(s,a) = Q(s,a) + U(s,a) \tag{4.9}$$

$$= \frac{W(s,a)}{N(s,a)} + c \cdot P(s,a) \cdot \frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)} \tag{4.10}$$

where c is a constant parameter controlling the level of exploration. Note that the primary distinction in our UCB score calculation compared to the naive approach (i.e., see Equation 4.7) is that the P values are generated by neural networks.

In the expansion step, the leaf node selected in the previous step will be expanded. First, the state s of the leaf node will be calculated as the chemical product of the state and action molecules represented by the edge which directs to this leaf node. We then find the action space $A(s)$ of the leaf node and calculate $P(s,a)$ values for all $a \in A(s)$ via the policy network, i.e., $P(s,\cdot) = f_\theta(s)$. The P values will be stored in the newly-added outgoing edges from the selected leaf node.

Meanwhile, the selected leaf node will be evaluated, which is done by the rollout step. This step aims to get a value for the selected leaf node. If the selected leaf node already represents a valid final product (i.e., a two-tail lipid in our case), then the value is simply the property score calculated by the property predictors. If the selected leaf node is not a valid product but an intermediate product, we conduct the rollout. The rollout means performing random actions until we reach the end of the play (i.e., until we generate a two-tail lipid). This randomly generated product will be evaluated by the property predictor and this property score will be sent back to act as the value of the selected leaf node. Note that this random generation path is only used for evaluating the leaf node during this simulation and won't be added to the search tree.

Once we have the value of the selected leaf node, we perform the last step of the simulation, backpropagation. The value will be backpropagated along the chosen path to update action values Q.

The detailed algorithm can be found in Appendix A.1.

4.4.2 Policy Network Training

As is mentioned before, the visit counts of all state-action pairs that appear in the tree search are recorded to be the training data of the policy network. However, the search data we collect are highly imbalanced in the sense that there will be a lot more state-action pairs in later levels. The number of state-action pairs from the first level (i.e., when state is the empty state and actions are the chosen head building blocks) is very limited, being the number of

different head building blocks that are expanded by the root node. Meanwhile, search data from later levels will be very sparse in the sense that the majority of the state-action pairs will have zero visit count. In order to better utilize our limited data, we propose a customized method for training the policy network.

Let $f(s,a)$ denote the naive policy network output (before softmax) for the state-action pair (s,a), $p(s,a)$ denote the corresponding predicted prior (which will be used in the UCB score calculation), and $\pi(s,a)$ be the search probability. We here adopt the search probability definition as proposed in the AlphaZero algorithm (Silver et al., b).

$$p(s,a_1) = \frac{e^{f(s,a_1)}}{\sum_a e^{f(s,a)}} \tag{4.11}$$

$$\pi(s,a_1) = \frac{N(s,a_1)^{\frac{1}{\tau}}}{\sum_a N(s,a)^{\frac{1}{\tau}}} \tag{4.12}$$

where τ is a temperature parameter controlling the level of exploration. When τ tends to infinity, the search probability is the same as random selections. When τ is small, say, when $\tau = 1$, the search probability strictly follows the actual visit count distribution. The objective of the policy network training is to maximize the similarity of $p(s,a)$ and $\pi(s,a)$.

Now, suppose we take two state-action pairs, (s,a_1) and (s,a_2), at a time. We apply the log-ratio and get:

$$\log \frac{p(s,a_1)}{p(s,a_2)} = f(s,a_1) - f(s,a_2) \tag{4.13}$$

$$\log \frac{\pi(s,a_1)}{\pi(s,a_2)} = \frac{1}{\tau}\log N(s,a_1) - \frac{1}{\tau}\log N(s,a_2) \tag{4.14}$$

The previous objective is equivalent to maximize the similarity of $f(s,a_1) - f(s,a_2)$ and $\frac{1}{\tau}\log N(s,a_1) - \frac{1}{\tau}\log N(s,a_2)$.

We therefore customize the loss function to be the error between $f(s,a_1) - f(s,a_2)$ and their corresponding $\frac{1}{\tau}\log N(s,a_1) - \frac{1}{\tau}\log N(s,a_2)$, for any two state-action pairs, (s,a_1) and (s,a_2). This can be considered as a regression problem, MAE or MSE loss may be applied.

4.5 Evaluation Metrics

The primary goal of our MCTS-based generative model is to produce ionizable lipids that are synthesizable. We first assess the model's capacity to generate ionizable lipids, and then examine the quality and synthesizability of the generated compounds.

4.5.1 Quality Evaluation

The primary criterion for evaluating our generated products is the ionizable lipid rate. This is determined by the proportion of unique generated products predicted to be both lipid-like and ionizable according to our property predictors. Our goal is to maximize this ratio. Additionally, we assess the validity of all generated ionizable lipids. Validity measures the proportion of the generated products that can be parsed by the chemoinformatics software RDKit (RDKit, nd).

4.5.2 Retrosynthesis Evaluation

While the ideal method to validate the synthesis pathway of our generated products would be through experimental testing in a laboratory, practical constraints currently prevent us from doing so. Consequently, we rely on in silico evaluations to estimate synthesizability. We first calculate the synthetic accessibility score (SA score) to predict synthesizability in an automated manner (Ertl and Schuffenhauer, 2009). Additionally, we seek to validate our proposed synthesis pathways using Syntheseus (Maziarz et al., 2023), a python package for retrosynthetic planning.

SA Score

The SA score is calculated through a combination of fragment contributions and a complexity penalty, derived from the analysis of over a million synthesized chemicals (Ertl and Schuffenhauer, 2009). Fragment contributions refer to the ease or difficulty of synthesizing specific molecular fragments based on their prevalence and synthetic history in known compounds. Molecular complexity is evaluated by considering elements such as the presence of large rings, non-standard ring fusions, stereochemical complexity, and overall molecule size. The resulting SA score provides a metric for synthetic accessibility, ranging from 1 (indicating easy synthesis) to 10 (indicating high difficulty). However, while the SA score differentiates between feasible and infeasible molecules to some extent, it does not provide specific insights into the actual synthesis pathways.

Retrosynthesis Planning via Syntheseus

Syntheseus operates by recursively decomposing a target molecule into increasingly simpler molecules through a backward reaction prediction model, continuing until it identifies a set of purchasable or commonly known building blocks (Maziarz et al., 2023). In our experiments, we validate the proposed synthesis pathway for a generated two-tail lipid. Specifically, we

assess whether this final product can be synthesized from an intermediary one-tail lipid and a purchasable tail building block. Additionally, we verify if the intermediate product can be constructed using provided head and tail building blocks. For these predictions, we employ the Molecule Edit Graph Attention Network (MEGAN) backward reaction prediction model (Sacha et al., 2020).

Chapter 5

Experiments and Results

In this chapter, we present a comprehensive analysis of the experiments conducted to evaluate the efficacy of our MCTS-based generative models for ionizable lipid generation. We first present the performance of our lipid property predictor, which set the foundation for our MCTS-based generative models. We then compare the generation results of the naive MCTS approach with our proposed policy network guided MCTS approach.

5.1 Lipid Property Predictor

Lipid property predictors are essential in our lipid generation process, evaluating the potential of generated molecules to function as ionizable lipids. These predictors generate a property score for each product, which is subsequently used to guide the MCTS in future simulations. Our lipid property prediction framework comprises two key components: a lipid classifier that provides a binary classification to determine if a product is lipid-like, and an ionizability predictor that offers a binary score assessing the ionizability of the product. Together, these tools ensure that the MCTS is strategically directed towards synthesizing molecules with the desired lipid characteristics. Detailed information of the property predictors has been discussed in Section 4.1.

As previously mentioned, our training dataset for the lipid classifier comprises approximately 180 000 lipid samples and an equal number of non-lipid samples. Utilizing the Chemprop graph neural network framework (Yang et al., 2019), our classifier features three message passing layers that integrate molecular features, complemented by two feed-forward layers dedicated to predicting molecular properties. After training for a single epoch, our model demonstrates exceptional performance, achieving both Receiver Operating Characteristic Area Under the Curve (ROC-AUC) and Precision-Recall Area Under the Curve

Table 5.1 Test performance of MolGpKa on pKa prediction.

	Acids			Bases		
	R^2	MAE	RMSE	R^2	MAE	RMSE
Organic Oxygen Acids and Nitrogen Bases	0.97	0.29	0.45	0.94	0.5	0.81
Novartis	0.75	0.85	1.29	0.75	0.79	1.06

(PR-AUC) scores above 0.9999. Together, these metrics, along with a test accuracy of 99.89%, underscore the robustness and predictive accuracy of our lipid classifier.

As discussed in Section 4.1.2, our approach to ionizability prediction leverages the MolGpKa module (Pan et al., 2021). This module identifies ionizable groups within a target molecule and calculates their corresponding pKa values. Utilizing these pKa values, we then determine the net charge of the target molecule at specific pH levels. Our ionizability predictions are specifically tailored based on two filtering criteria, at pH levels of 5 and 7.4, respectively. According to (Pan et al., 2021), the MolGpKa predictor has undergone rigorous testing with the Organic Oxygen Acids and Nitrogen Bases set (Yu et al., 2010) and the Novartis set (Baltruschat and Czodrowski, 2020). Its performance is evaluated using three metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and the coefficient of determination (R^2). Detailed results of these evaluations are presented in Table 5.1, highlighting the robustness and accuracy of the MolGpKa module in predicting pKa values and thus justifying the credibility of our ionizability predictions.

Combining the capabilities of our lipid classifier and ionizability predictor, our property predictors provide reliable evaluations for our generated products. During the property scoring phase in subsequent MCTS simulations, we use a binary indicator (0 or 1) to denote the ionizability of a molecule. However, for the lipid classifier, we utilize the network's raw output scores, which range within $(0, 1)$, before applying binarization to assign property scores. As a result, the combined property score from both the lipid classifier and ionizability prediction ranges within $(0, 2)$, with most scores expected to be close to 0, 1, or 2. A property score approaching 2 suggests the molecule is likely an ionizable lipid; a score near 1 indicates the molecule may either be non-lipid-like or non-ionizable; and a score close to 0 implies the molecule is neither lipid-like nor ionizable. This scoring system enhances our ability to categorize and prioritize molecules efficiently in our search for valid ionizable lipids.

5.2 Monte Carlo Tree Search for Lipid Generation

In this section, we delve into the practical application of MCTS-based generative models for ionizable lipid generation, utilizing the robust framework developed in previous chapters. Our experimental investigation focuses on comparing the performance of the naive MCTS approach, derived from SyntheMol (Swanson et al., 2024) and served as the baseline, with our proposed policy network guided MCTS approach. We demonstrate the enhanced efficiency and effectiveness of the guided MCTS approach in identifying and synthesizing high-potential ionizable lipids.

5.2.1 Experimental Setups

In this section, we detail the specific subset of the lipid building block dataset utilized in our experiments, along with the configurations of both MCTS-based generative models. These configurations establish the experimental settings for all subsequent experiments discussed in this chapter. Additionally, we outline the computing resources employed to conduct these experiments.

Lipid Building Block Dataset

As mentioned in Chapter 3, we have constructed a lipid building block dataset consisting of over 2.7 million lipid head building blocks and 5310 lipid tail building blocks. Due to the extensive size of the head building block dataset, directly incorporating all entries into the MCTS would result in a predominantly exploratory behavior akin to random selections. To mitigate this, we have curated a subset of approximately 12 000 head building blocks to define our actual head search space for the MCTS, while the entire tail building block dataset is utilized in subsequent levels of the search.

Both the naive MCTS approach and the policy network-guided MCTS approach utilize these same datasets. However, for the evaluation of the policy network in the guided MCTS approach, an additional set of 200 testing head building blocks is employed. Importantly, this testing head search space does not overlap with the head search space used during the training of the policy network, ensuring the integrity and validity of our testing protocols.

Naive MCTS Configuration

We have imposed a constraint on the maximum number of child node expansions to 2 000, meaning that the MCTS explores a head search space of this size. The MCTS is executed over 10 000 simulations to ensure comprehensive analysis of the generated lipid products.

Additionally, the exploration weight c used in the UCB score calculation, as detailed in Equation 4.7, is set at 10. This parameter setting is designed to balance the exploration and exploitation phases of the search process effectively.

Guided MCTS Configuration

We operate the guided MCTS across 10 iterations, with each iteration involving a dual phase of running the MCTS to accumulate search data and subsequently using this data to train the policy network. In the following iteration, the newly updated policy network is deployed. For every iteration, the MCTS is executed 10 times, each exploring a head space of 200. This setup allows the 10 runs of MCTS collectively to explore a total head search space size of 2 000, aligning with the head search space used in the naive MCTS approach.

Each MCTS runs 10 000 simulations to gather substantial search data for effective training of the policy network. The data from these 10 MCTS runs are pooled to train the policy network, and the accumulated generated products from these runs are analyzed as the iteration's output. The policy network undergoes 20 epochs of training in each iteration to optimize its performance. Moreover, the exploration weight c used in the UCB score calculation, as outlined in Equation 4.9, is set at 20.

Policy Network Configuration

The policy network in our study processes state-action pairs, where each state and action corresponds to individual molecules—the state being the current molecule and the action being the next building block molecule selected. Each molecule is represented using a Morgan fingerprint with 1024-bit binary digit (Schneider et al., 2015), and the fingerprints of both the state and action molecules are concatenated to form a 2048-bit feature vector, serving as the input to the policy network.

Our policy network architecture consists of four linear layers, each followed by a ReLU activation function (Fukushima, 1969) and dropout layers with a dropout rate of 0.5 (Wan et al., 2013) to prevent overfitting. The custom loss function used in the training of this network is elaborated upon in Section 4.4.2. We employ the Adam optimizer for network training, utilizing a learning rate of 0.001 (Kingma and Ba, 2017).

Computing Resources

Our computational setup includes a GPU server equipped with 8 Tesla P-100 GPUs, each featuring 16 GB of memory. It is important to note that within our experimental framework, only the MolGpKa module is configured to utilize GPU resources (Pan et al., 2021). All

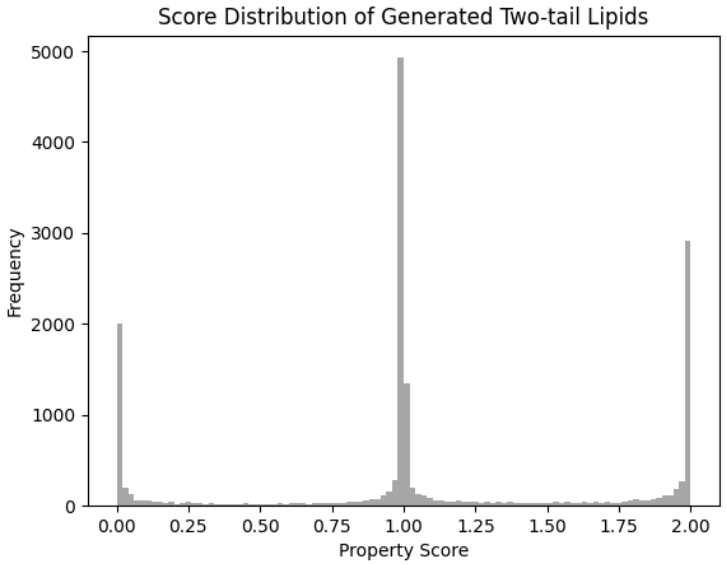


Fig. 5.1 Property score distribution of the generated products of the naive MCTS.

other components of our algorithms, including the MCTS simulations and the neural network training for the policy network, are designed to run efficiently on CPU.

5.2.2 Generation Results and Discussions

Our primary objective is to generate products that are potent candidates for ionizable lipids, meaning we aim for our model to produce molecules with high property scores. To evaluate the effectiveness of our approach, we begin by examining the products generated through the naive MCTS approach. This analysis serves as a baseline for comparing the performance enhancements achieved with the guided MCTS approach, thereby providing a clear benchmark for assessing improvements in generating high-potential ionizable lipid candidates.

Figure 5.1 displays the score distribution of all two-tail lipids generated through the naive MCTS. The property score combines a lipid classifier score, which typically hovers near 0 or 1, and a binary ionizability score. A property score approaching 2 typically identifies the molecule as a likely ionizable lipid; a score around 1 indicates that the molecule may either be non-lipid-like or non-ionizable; and a score close to 0 suggests that the molecule is neither lipid-like nor ionizable. From the figure, it is evident that the majority of the products are classified as either non-lipid-like or non-ionizable, with only a small fraction identified as potential ionizable lipids.

Derailed statistics of the generated products are presented in Table 5.2. After conducting 10 000 simulations, a total of 16 477 two-tail lipids were generated. We observe that only

Table 5.2 Overview of generated products from naive MCTS simulations.

	Generated Product Number	Ratio
Lipid-like & Ionizable	4513	0.2739
Lipid-like & Not Ionizable	6198	0.3762
Not Lipid-like & Ionizable	2696	0.1636
Not Lipid-like & Not Ionizable	3070	0.1863
Total Product Number	16477	/

4513 of the 16 477 generated products are predicted to be ionizable lipids, providing an ionizable lipid rate at 0.2739.

It is important to note that the number of generated products exceeds the number of simulations, which results from the possibility of multiple products being formed during a single simulation. This occurs particularly when two candidate molecules—specifically, a state molecule and an action molecule—undergo reactions that yield distinct products. Such multiplicity in product formation is plausible when candidate molecules feature functional groups that recur within the same molecule. When these functional groups engage in reactions with other molecules, the varied reactive possibilities lead to the generation of different products. This phenomenon underscores the complexity and diversity of chemical reactions explored during the simulations. For the purpose of generating more ionizable lipid candidates, we include all the possible products in our results. However, due to the specific mechanism of the guided MCTS approach, only one of the multiple products can be considered. In practice, a random selection process is employed during guided MCTS simulations to choose one of the possible products. This approach not only simplifies the decision-making process within the simulation but also introduces an element of stochasticity that reflects real-world screening and selection scenarios.

We now turn our attention to the outcomes of the guided MCTS approach. By comparing these results with those from the naive approach, we seek to illustrate the enhanced effectiveness of the guided MCTS in producing ionizable lipids. We begin by reviewing the training results of the policy network and then analyze the products generated during both training and testing simulations.

Figure 5.2 presents the training loss curve of the policy network plotted against the number of training epochs. The data from each iteration of the training process is represented on the same graph but distinguished by different colors for clarity. While the policy network undergoes training for 20 epochs during each iteration, the figure selectively displays the loss curve only up to the epoch at which the best-performing model state is achieved. Consequently, the total number of epochs represented in the figure is fewer than the expected

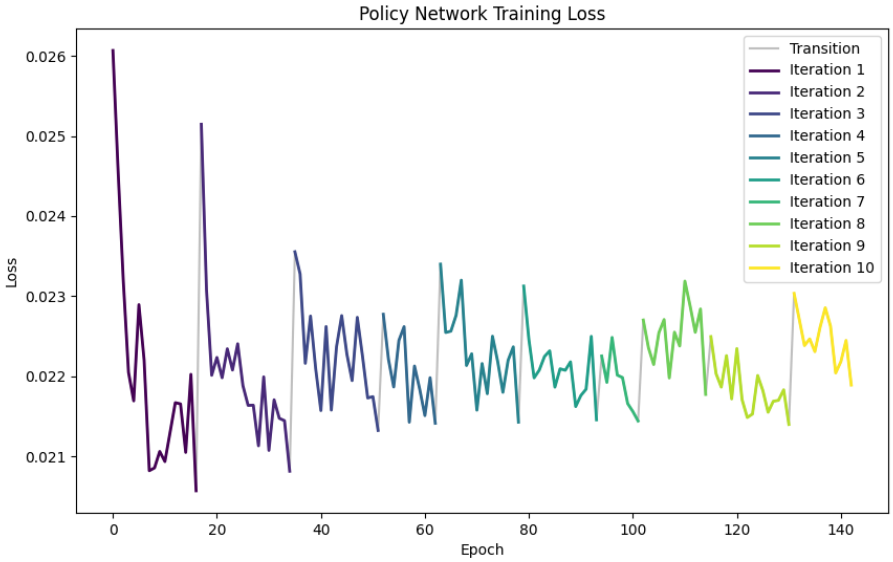


Fig. 5.2 Policy network training loss against training epochs.

200, illustrating the points at which optimal performance was reached rather than the entire training span.

Recall that each iteration of the MCTS simulations explores a different segment of the lipid head search space, each consisting of 2 000 entities, while the total lipid head search space utilized for training encompasses approximately 12 000 entities. Consequently, while the lipid head search spaces in the initial iterations are largely distinct, they begin to overlap in the later iterations. This setup means that, particularly from the seventh iteration onward, some search data may reappear, having been included in earlier iterations.

Each iteration of the policy network training starts with a relatively high loss, indicating the model's initial unfamiliarity with a new and largely distinct dataset, particularly notable in the first few iterations (i.e., iterations 1-3). This initial high loss is anticipated as the model encounters new lipid head entities previously unseen during training. The rapid decline in loss during the early epochs across these initial iterations highlights the network's capability to quickly adapt to novel datasets. This swift reduction in loss signifies that the policy network is effectively learning and accommodating the unique characteristics of each segment of the search space. After the initial steep learning curve observed in the first three iterations, the loss fluctuations during iterations 4 through 6 begin to show a trend towards stabilization, suggesting that the network is refining its parameters and optimizing its performance relative to the given datasets. In later iterations (e.g., iterations 7-10), these fluctuations become even smaller, likely due to the overlap in data. This smaller variation also implies that the network

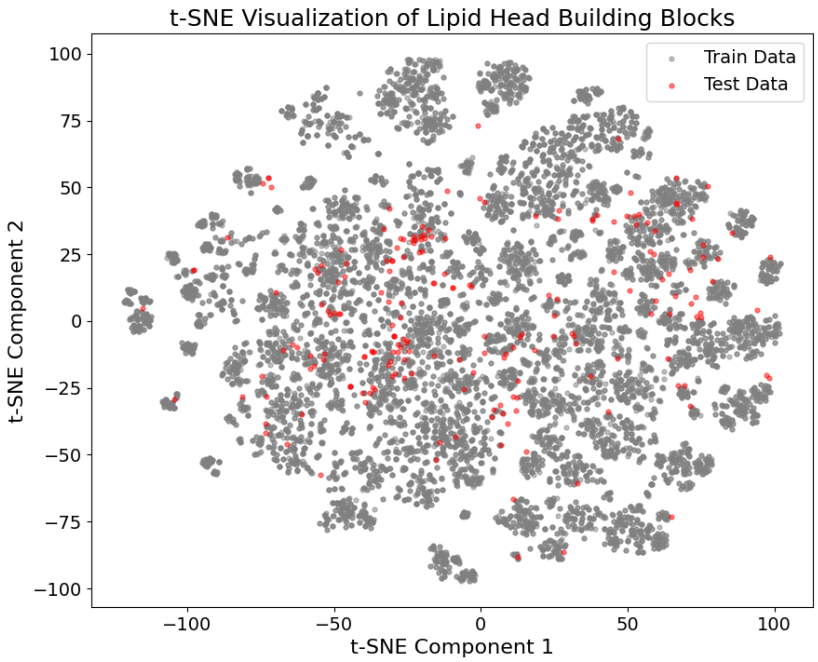


Fig. 5.3 A t-SNE visualization of the training and testing lipid head building block molecules.

has successfully captured the underlying patterns common to the datasets it has been exposed to.

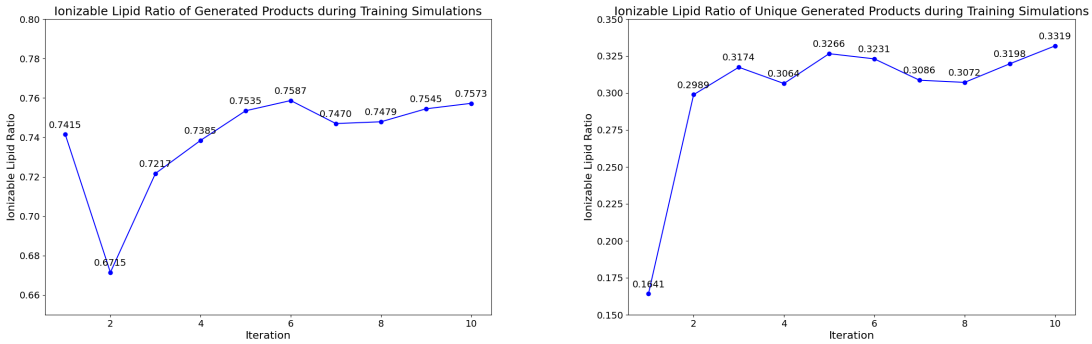
The variability in final loss values across iterations points to differences in the network's ability to generalize across datasets, with some datasets yielding better performance as evidenced by lower final loss values. However, the overall decreasing trend in training loss underscores the reliability and effectiveness of our policy network.

Next, we will analyze the products generated by the MCTS simulations guided by the policy network trained across various iterations. We aim to evaluate the efficiency of the policy network in enhancing the generation process and improving the quality of the lipid molecules produced.

We analyze the generated products from both training and testing simulations, where training simulations provide search data for training the policy network and testing simulations employ a separate test set of head building block molecules. The primary distinction between these simulations lies in their respective head building block search spaces. To assess the representativeness of both datasets, we conduct a t-Stochastic Neighbor Embedding (t-SNE) visualization (van der Maaten and Hinton, 2008) of the 1024-bit binary digit Morgan fingerprint representations of the head molecules, as shown in Figure 5.3. This visualization

reveals that, despite its smaller size, the testing head dataset comprehensively covers most of the molecular space encompassed by the training dataset.

However, it is important to note that the selection of head molecules significantly influences the quality of the generated products. Consequently, differences in the head search space can lead to substantial variations in the overall quality of the products. Nonetheless, observing the trend in quality changes within a specific head set, as guided by different iterations of the policy network, can still effectively illustrate the efficiency and impact of policy network training.



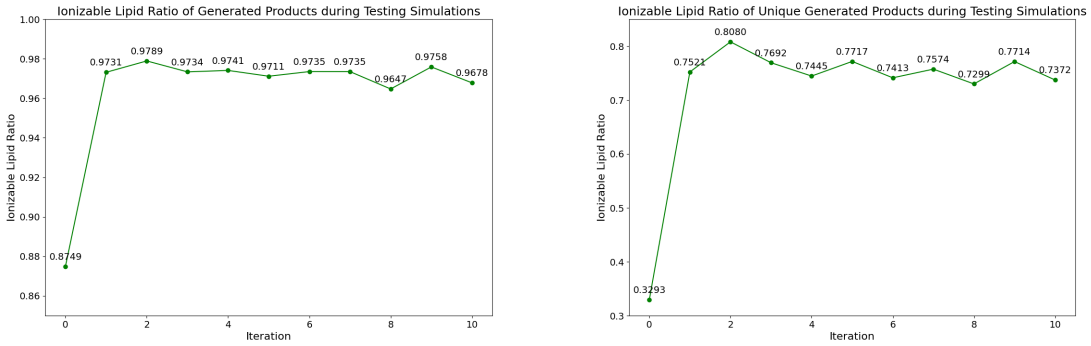
(a) Ionizable lipid rate of total products

(b) Ionizable lipid rate of unique products

Fig. 5.4 Ionizable lipid rate of the generated products during training MCTS simulations

We begin by examining the generated products during training simulations. Figure 5.4a illustrates the ionizable lipid rate across training iterations for all generated products, including duplicates, while Figure 5.4b depicts the ionizable lipid rate for unique products only. The initial results in iteration 1 are derived from MCTS simulations guided by a randomly initialized policy network, whereas the results from iteration 2 onward are influenced by successively trained instances of the policy network. Analysis of Figure 5.4a indicates that the ionizable lipid rate surpasses that of the randomly initialized simulations beginning with the fifth iteration, post four training sessions of the policy network. However, this improvement is not markedly clear. In contrast, a distinct increase in the ionizable lipid rate is observed from the very first training session when analyzing the unique generated products, as shown in Figure 5.4b. Since our primary interest lies in the quality of unique generated products, this data underscores a significant enhancement in quality attributable to the training of the policy network, affirming its efficacy in refining the generation process.

Figure 5.5 showcases the products generated during testing simulations, each guided by policy networks trained across different iterations. Figure 5.5a illustrates the ionizable lipid rate for all generated products, inclusive of duplicates, while Figure 5.5b focuses



(a) Ionizable lipid rate of total products

(b) Ionizable lipid rate of unique products

Fig. 5.5 Ionizable lipid rate of the generated products during testing MCTS simulations

exclusively on the ionizable lipid rate for unique products only. The initial iteration (Iteration 0) features products generated by MCTS simulations directed by a randomly initialized policy network, with subsequent iterations using progressively trained policy networks. Both figures reveal a notable increase in the ionizable lipid rate following the first iteration of policy network training. Subsequent iterations show the rate stabilizing, with only minor fluctuations observed. Specifically, the ionizable lipid rate varies between 0.96 and 0.98 for duplicated products, and oscillates between 0.73 and 0.8 for unique products. Comparatively, these rates significantly surpass the ionizable lipid rate of below 0.3 achieved by the naive MCTS approach, clearly demonstrating the superior performance of our model over the baseline.

To better illustrate the improvement in product quality, Figure 5.6 shows the shift in the distribution of property scores for unique generated products during testing simulations across iterations. The shape of each violin plot reflects the distribution of property scores within that iteration, with the width at any given point along the y-axis indicating the density of data points at that score. In the initial iteration (iteration 0), the distribution is relatively broad, with a significant spread from low to high property scores. This broad distribution suggests that the initial exploration of the chemical space is varied, capturing a wide range of property scores. We observe a notable shift in the score distribution during the first two iterations, which aligns with the increase in the ionizable lipid rate shown in Figure 5.5b, indicating that the model is actively learning and adapting to optimize its search strategy. In the later iterations, the majority of the scores cluster near 2, with a smaller fraction close to 1, and very few near 0. We do not detect any significant changes in the score distribution in these later iterations, suggesting that the network is converging towards a relatively stable

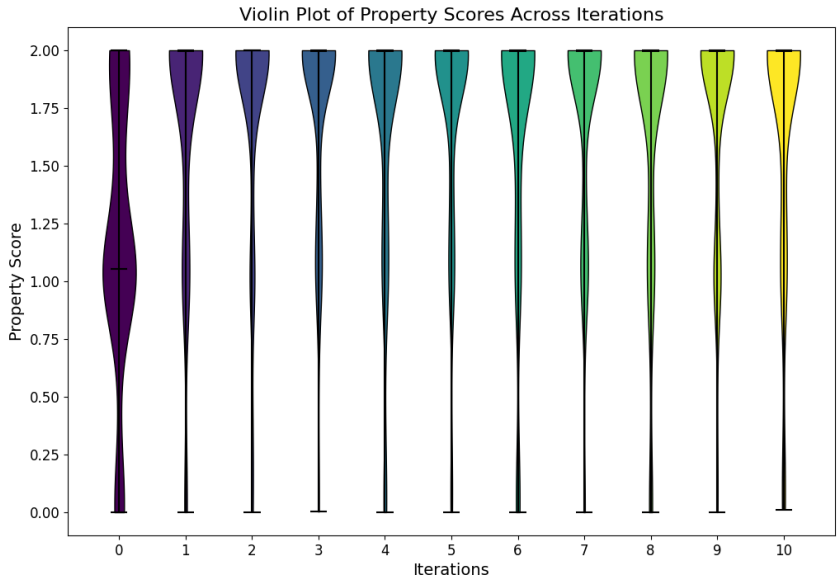


Fig. 5.6 Property score distribution of unique generated products during testing simulations.

product quality. This stability is consistent with the small fluctuations in the ionizable lipid rate observed in Figure 5.5b.

A detailed overview of the products generated during testing simulations is provided in Table 5.3. It lists the total number of generated products (including duplicates), the number of unique products, the number of unique ionizable lipids, and the number of unique ionizable lipids produced in the first 500 generations. Previous discussions have highlighted the high ionizable lipid rates—calculated as the ratio of unique ionizable lipids to unique generated products—after training the policy network.

Analysis of the data shows a relatively low uniqueness rate among the generated products, which can be attributed to the search mechanism of MCTS. Particularly in later simulations, MCTS tends to favor previously visited actions and paths known to yield high rewards, which could lead to a reduction in uniqueness rate as more simulations are conducted. Consequently, the uniqueness rate does not serve as a reliable metric for evaluating the sampling efficiency of our MCTS-based generative models.

The last column of the table shows the number of unique ionizable lipids generated within the first 500 generations. Given that we conducted 10 000 testing simulations in total, it is noteworthy that over 70% of the final unique ionizable lipids were produced during these early stages. This observation underscores the effectiveness of the policy network guided MCTS in efficiently exploring new possibilities early in the simulation process.

Table 5.3 Overview of generated products during testing simulations across iterations

Iteration	Total Products	Unique Products	Unique Ionizable Lipids	Unique Ionizable Lipids (first 500 generation)
0	9204	167	55	44
1	9079	117	88	78
2	9091	125	101	80
3	9082	130	100	81
4	9104	137	102	72
5	9114	127	98	77
6	9169	143	106	78
7	9123	136	103	80
8	9121	137	100	78
9	9109	140	108	75
10	9132	137	101	74

Table 5.4 Retrosynthesis evaluation of generated products and generated synthesis pathway.

	Validity	Average SA Score	Retrosynthesis Rate by Syntheseus	No. Ionizable Lipids Evaluated
Naive MCTS	1	4.77	0.0363	4513
Guided MCTS Training	1	4.62	0.0777	5058
Guided MCTS Testing	1	4.24	0.0294	545

5.3 Retrosynthesis Evaluation

Table 5.4 presents the evaluation results of our generated ionizable lipids and their corresponding synthesis pathways. We consider the generated ionizable lipids during all the simulations of both the naive MCTS approach and the guided MCTS approach. To ensure a broader and more generalized evaluation, we consider all products generated during both training and testing phases across all iterations of the guided MCTS approach. The number of unique ionizable lipids to be evaluated is presented in the last column of the table. The first column of the table shows that the validity rate for all generated molecules is 1, indicating that all molecules can be successfully parsed by RDKit (RDKit, nd).

The SA score, which ranges from 1 to 10 with lower values indicating easier synthesis (Ertl and Schuffenhauer, 2009), shows similar results for both the naive and guided MCTS approaches, with the guided approach slightly outperforming the naive approach. While the SA score justifies the synthesizability of the generate products to some extent, we are aware that they provide no insights on our synthesis pathways.

Using Syntheseus, we validate the synthesis pathways identified by the MCTS, as discussed in Section 4.5.2. We assess whether the generated two-tail lipids can be synthesized from an intermediary one-tail lipid and a purchasable tail building block, and whether this intermediate product can be constructed using the provided head and tail building blocks. Syntheseus identifies valid synthesis routes based on the input product and building blocks. We record the proportion of generated products and their corresponding synthesis pathways that are validated by this retrosynthesis tool.

We observe that the retrosynthesis rate validated by Syntheseus is quite low, with none of the cases exceeding 8%. This low rate may result from the unique characteristics of the lipid building block dataset and the reaction templates we adopted from the SyntheMol project, which are tailored for the Enamine REAL space (Grygorenko et al., 2020; Swanson et al., 2024). For retrosynthesis planning, we utilize the MEGAN backward reaction prediction model trained on the USPTO-50k dataset (Sacha et al., 2020), which was collected by (Lowe, 2012) and classified into 10 reaction types by (Schneider et al., 2016). Our lipid building block molecules may differ significantly from the reactants in USPTO reactions, and our reaction templates may not align well with those in the USPTO dataset. Additionally, a lot of the reaction templates involve using linker molecules, while such reactions might not be present in USPTO. These mismatches make retrosynthesis planning challenging and contribute to the low retrosynthesis rate.

Selected examples of generated ionizable lipids, along with their synthesis pathways validated by the retrosynthesis tool, are presented in Appendix C.

Chapter 6

Discussion

In this dissertation, we have explored the Monte Carlo tree search (MCTS) based generative models for ionizable lipid generation. We constructed a lipid building block dataset, featuring purchasable ionizable lipid heads and tails, which is well-suited for future lipid generation projects. We developed two lipid property predictors: a lipid classifier and an ionizability predictor, both designed to accurately assess whether a candidate molecule is lipid-like or ionizable. Adapting the SyntheMol approach, originally utilized for antibiotic discovery, we tailored this method for lipid generation to serve as our baseline model. We further innovated by developing a policy network guided MCTS-based generative model which is capable of producing high-quality ionizable lipids with available synthesis paths, outperforming our baseline. Our achievements indicate that this project offers a promising direction for ionizable lipid generation, also contributing to the broader field of drug delivery. However, it is important to acknowledge that the project has its limitations, which are discussed alongside future work in this chapter.

6.1 Limitations

In this section, we discuss several key limitations inherent in our current generative model and retrosynthesis validation approaches for ionizable lipid generation.

6.1.1 Generative Model

Our generative model heavily depends on the reaction templates we use to conduct forward reaction predictions. We have adopted the same 13 reactions utilized by SyntheMol, which are responsible for producing 93.9% of the molecules within the Enamine REAL space (Grygorenko et al., 2020; Swanson et al., 2024). These reactions were chosen because they

are common and widely applicable, and the functional groups involved in these reactions are also present in our lipid building blocks. However, it is crucial to recognize that our search space, which consists of lipid building blocks, differs from the Enamine REAL space. Consequently, some of the reactions we currently use may not have been empirically validated with lipid building block molecules. Furthermore, it is possible that our existing templates do not encompass reactions that are typically employed in practical lipid generation. This discrepancy highlights a potential limitation in the direct applicability of our model to lipid synthesis and underscores the need for further validation and potential adaptation of reaction templates to better suit lipid generation.

Additionally, while the MCTS is designed to efficiently explore combinatorial chemical spaces, the actual scope of exploration is significantly constrained by the number of child nodes involved in each expansion. Specifically, in our model, the exploration potential at the root node, which navigates the head search space, is directly limited by the number of node expansion at this initial stage. Given that our lipid head building block dataset comprises a vast pool of 2.7 million molecules, it is logistically infeasible to conduct a thorough exploration of this entire head search space. This limitation inherently restricts our ability to fully utilize the diversity available in the dataset, leading to a critical bottleneck in the current configuration of our MCTS framework.

6.1.2 Retrosynthesis Validation

The most definitive method of validating any synthesized molecule is through empirical laboratory experiments. However, such experiments are not feasible in our project due to practical constraints such as high costs, time requirements, and lack of necessary equipment and materials. This restricts our ability to fully confirm the efficacy and safety of the synthesized compounds outside of a theoretical or simulated environment.

Secondly, there is a notable gap in our knowledge and access to sophisticated computer-aided retrosynthesis tools that are specifically tailored to our research scenario. While there are general tools available, they may not fully accommodate the unique requirements or the specific complexities of ionizable lipid generation. This hampers our ability to perform detailed and accurate retrosynthesis analysis, which is crucial for understanding how a molecule can be synthesized in the most efficient and feasible manner.

Additionally, a significant limitation in our analysis is the absence of a baseline model for retrosynthesis analysis. Currently, we do not employ a standard generative method that disregards synthesis pathways during molecule generation as a comparative measure. Including such a baseline could be beneficial, as it would allow us to compare the retrosynthesis rate of molecules generated without considering their synthetic pathways against those

produced by our MCTS-based methods that integrate synthesis considerations. Establishing this baseline would enhance the robustness of our study and provide a clearer understanding of the improvements our methods offer over traditional molecular generation techniques.

6.2 Future Work

Based on the limitations discussed previously, we outline future perspectives aimed at enhancing our generative models and achieving effective retrosynthesis validation.

6.2.1 Generative Model

To address the limitations identified in our current generative model, particularly concerning the reaction templates and the restricted search space of the MCTS, we propose the following directions for future research. Additionally, while ionizable lipids are primarily utilized in mRNA delivery, the transfection efficiency of these lipids is also critically important and represents a key area of interest. Optimizing this aspect could significantly impact the effectiveness of lipid-mediated drug delivery systems.

Reaction Template Validation and Expansion If condition allowed, we may conduct experimental validations of the existing reaction templates with lipid building blocks to ensure their applicability and efficiency in lipid synthesis. It will be helpful to develop and integrate additional reaction templates that are specific to lipid chemistry. Collaborating with chemists to identify and codify reactions commonly used in lipid generation could expand the utility and accuracy of our model.

Enhancing MCTS Exploration Capabilities While it's difficult to explore a search space with a vast size of 2.7 million, preprocessing the lipid head building block dataset to refine the search space can make exploration more manageable and targeted. One potential strategy involves refining the head search space by leveraging sophisticated biochemical characteristics of the head building blocks, identifying those most promising for ionizable lipid generation. This process would require advanced chemical knowledge. Another approach is to develop a hierarchical searching strategy. This would involve initially grouping head building blocks based on shared structural features or similar biomedical properties. MCTS could then be applied to explore these groups systematically, delving into detailed exploration of individual molecules only within groups that demonstrate the most promise based on preliminary results. Besides, if computing resource permits, conducting more extensive tree searches across a broader array of head building block subsets would inherently increase the likelihood of discovering high-potential candidates.

Enhancing Transfection Efficiency in Ionizable Lipids A straightforward method to integrate transfection efficiency into our research is to filter the final set of generated ionizable lipids based on their transfection performance, retaining only those that exhibit high performance. To further improve the transfection efficiency of our synthesized lipids, we can incorporate a transfection efficiency prediction model into our existing suite of lipid property predictors. This additional model would work alongside the current lipid classifier and ionizability predictor to guide the MCTS process more effectively. Existing deep learning models designed for evaluating transfection efficiency may be utilized (Xu et al., 2024).

6.2.2 Retrosynthesis Validation

To address the limitations regarding retrosynthesis validation identified in the current study, the following future work directions are proposed.

Laboratory Synthesis Experiments To address the limitation of not having practical synthesis validations, future research should prioritize establishing collaborations with chemical laboratories. This will enable empirical testing and validation of the synthesized molecules, providing a direct assessment of their practical viability and safety. This step is crucial for transitioning from theoretical designs to real-world applications, ensuring that the molecules not only exist on paper but can also be effectively and safely synthesized in a lab environment.

Development of Specialized Retrosynthesis Tools There is a pressing need to develop or customize computer-aided retrosynthesis tools specifically for ionizable lipid generation. Traditional retrosynthesis planning primarily utilizes backward reaction prediction models. Enhancing traditional retrosynthesis with backward reaction algorithms specifically optimized for the complex structures of ionizable lipids could improve their synthesis. These algorithms should efficiently manage large molecules with multiple functional groups and intricate stereochemistry. Optimizations could include advanced heuristics that prioritize certain chemical bonds or functional groups based on their reactivity and synthetic feasibility under standard laboratory conditions.

Incorporation of a Comparative Baseline Model To robustly evaluate the effectiveness of our generative methods, future studies should include a baseline model that does not consider synthesis pathways during molecule generation. This would enable a comparative analysis of retrosynthesis rates, demonstrating the benefits of integrating synthesis considerations into the generative process. By comparing our methods against a traditional generative approach, we can quantitatively assess improvements in synthesizability.

References

Aimo, L., Liechti, R., Hyka-Nouspikel, N., Niknejad, A., Gleizes, A., Götz, L., Kuznetsov, D., David, F. P. A., van der Goot, F. G., Riezman, H., Bougueleret, L., Xenarios, I., and Bridge, A. (2015). The swisslipids knowledgebase for lipid biology. *Bioinformatics*, 31(17):2860–2866.

Akinc, A., Maier, M. A., Manoharan, M., Fitzgerald, K., Jayaraman, M., Barros, S., Ansell, S., Du, X., Hope, M. J., Madden, T. D., Mui, B. L., Semple, S. C., Tam, Y. K., Ciufolini, M., Witzigmann, D., Kulkarni, J. A., van der Meel, R., and Cullis, P. R. (2019). The onpattro story and the clinical translation of nanomedicines containing nucleic acid-based drugs. 14(12):1084–1087.

Assouel, R., Ahmed, M., Segler, M. H. S., Saffari, A., and Bengio, Y. (2018). Defactor: Differentiable edge factorization-based probabilistic graph generation. *CoRR*, abs/1811.09766.

Baltruschat, M. and Czodrowski, P. (2020). Machine learning meets pka. *F1000Research*, 9(Chem Inf Sci):113. version 2; peer review: 2 approved.

Bian, Y., Wang, J., Jun, J. J., and Xie, X.-Q. (2019). Deep convolutional generative adversarial network (dcGAN) models for screening and design of small molecules targeting cannabinoid receptors. 16(11):4451–4460. Publisher: American Chemical Society.

Bradshaw, J., Paige, B., Kusner, M. J., Segler, M. H. S., and Hernández-Lobato, J. M. (2019). A model to search for synthesizable molecules. *CoRR*, abs/1906.05221.

Bradshaw, J., Paige, B., Kusner, M. J., Segler, M. H. S., and Hernández-Lobato, J. M. (2020). Barking up the right tree: an approach to search over molecule synthesis dags.

Bronstein, M. M., Bruna, J., Cohen, T., and Velickovic, P. (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478.

Carrasco, M. J., Alishetty, S., Alameh, M.-G., Said, H., Wright, L., Paige, M., Soliman, O., Weissman, D., Cleveland, T. E., Grishaev, A., and Buschmann, M. D. (2021). Ionization and structural properties of mRNA lipid nanoparticles influence expression in intramuscular and intravascular administration. 4(1):956.

Coley, C. W., Green, W. H., and Jensen, K. F. (2018). Machine learning in computer-aided synthesis planning. 51(5):1281–1289. Publisher: American Chemical Society.

Coulom, R. (2006). Efficient selectivity and backup operators in monte-carlo tree search. In *Computers and Games*.

Degors, I. M. S., Wang, C., Rehman, Z. U., and Zuhorn, I. S. (2019). Carriers break barriers in drug delivery: Endocytosis and endosomal escape of gene delivery vectors. 52(7):1750–1760. Publisher: American Chemical Society.

Ertl, P. and Schuffenhauer, A. (2009). Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. 1(1):8.

Evers, M. J. W., Kulkarni, J. A., van der Meel, R., Cullis, P. R., Vader, P., and Schiffelers, R. M. (2018). State-of-the-art design and rapid-mixing production techniques of lipid nanoparticles for nucleic acid delivery. *Small Methods*, 2(9):1700375.

Eygeris, Y., Gupta, M., Kim, J., and Sahay, G. (2022). Chemistry of lipid nanoparticles for RNA delivery. 55(1):2–12. Publisher: American Chemical Society.

Fukushima, K. (1969). Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Trans. Syst. Sci. Cybern.*, 5:322–333.

Gao, W. and Coley, C. W. (2020). The synthesizability of molecules proposed by generative models.

Gao, W., Mercado, R., and Coley, C. W. (2021). Amortized tree generation for bottom-up synthesis planning and synthesizable molecular design. *CoRR*, abs/2110.06389.

Gaulton, A., Bellis, L. J., Bento, A. P., Chambers, J., Davies, M., Hersey, A., Light, Y., McGlinchey, S., Michalovich, D., Al-Lazikani, B., and Overington, J. P. (2012). ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Research*, 40(D1):D1100–D1107.

Gottipati, S. K., Sattarov, B., Niu, S., Pathak, Y., Wei, H., Liu, S., Thomas, K. M. J., Blackburn, S., Coley, C. W., Tang, J., Chandar, S., and Bengio, Y. (2020). Learning to navigate the synthetically accessible chemical space using reinforcement learning. *CoRR*, abs/2004.12485.

Grygorenko, O. O., Radchenko, D. S., Dziuba, I., Chuprina, A., Gubina, K. E., and Moroz, Y. S. (2020). Generating multibillion chemical space of readily accessible screening compounds. *iScience*, 23(11):101681. Published correction appears in iScience. 2020 Dec 04;23(12):101873. doi: 10.1016/j.isci.2020.101873.

Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. (2018). Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268–276.

Han, X., Zhang, H., Butowska, K., Swingle, K. L., Alameh, M.-G., Weissman, D., and Mitchell, M. J. (2021). An ionizable lipid toolbox for RNA delivery. 12(1):7233.

Hou, X., Zaks, T., Langer, R., and Dong, Y. (2021). Lipid nanoparticles for mRNA delivery. 6(12):1078–1094.

Howard, R. A. (1960). *Dynamic programming and Markov processes*. Technology Press of Massachusetts Institute of Technology.

Irwin, J. J., Tang, K. G., Young, J., Dandarchuluun, C., Wong, B. R., Khurelbaatar, M., Moroz, Y. S., Mayfield, J., and Sayle, R. A. (2020). ZINC20—a free ultralarge-scale chemical database for ligand discovery. *Journal of Chemical Information and Modeling*, 60(12):6065–6073. PMID: 33118813.

Irwin, R., Dimitriadis, S., He, J., and Bjerrum, E. J. (2022). Chemformer: a pre-trained transformer for computational chemistry. *Machine Learning: Science and Technology*, 3(1):015022.

Jin, W., Barzilay, R., and Jaakkola, T. S. (2020). Hierarchical generation of molecular graphs using structural motifs. *CoRR*, abs/2002.03230.

Kim, M., Jeong, M., Hur, S., Cho, Y., Park, J., Jung, H., Seo, Y., Woo, H. A., Nam, K. T., Lee, K., and Lee, H. (2021). Engineered ionizable lipid nanoparticles for targeted delivery of rna therapeutics into different types of cells in the liver. *Science Advances*, 7(9):eabf4398.

Kim, S., Thiessen, P. A., Bolton, E. E., Chen, J., Fu, G., Gindulyte, A., Han, L., He, J., He, S., Shoemaker, B. A., Wang, J., Yu, B., Zhang, J., and Bryant, S. H. (2016). Pubchem substance and compound databases. *Nucleic Acids Research*, 44(D1):D1202–D1213.

Kim, Y.-K. (2022). RNA therapy: rich history, various applications and unlimited future prospects. 54(4):455–465.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907.

Kocsis, L. and Szepesvari, C. (2006). Bandit based monte-carlo planning. In *European Conference on Machine Learning*.

Kong, B., Kim, Y., Kim, E. H., Suk, J. S., and Yang, Y. (2023). mrna: A promising platform for cancer immunotherapy. *Advanced Drug Delivery Reviews*, 199:114993.

Kularatne, R., Crist, R., and Stern, S. (2022). The future of tissue-targeted lipid nanoparticle-mediated nucleic acid delivery. *Pharmaceuticals*, 15:897.

Kusner, M. J., Paige, B., and Hernández-Lobato, J. M. (2017). Grammar variational autoencoder.

Li, B., Manan, R. S., Liang, S.-Q., Gordon, A., Jiang, A., Varley, A., Gao, G., Langer, R., Xue, W., and Anderson, D. (2023). Combinatorial design of nanoparticles for pulmonary mRNA delivery and genome editing. 41(10):1410–1415.

Liu, Q., Allamanis, M., Brockschmidt, M., and Gaunt, A. L. (2019). Constrained graph variational autoencoders for molecule design.

Lowe, D. (2012). *Extraction of chemical structures and reactions from the literature*. PhD thesis, University of Cambridge.

Luo, S., Shi, C., Xu, M., and Tang, J. (2021). Predicting molecular conformation via dynamic graph score matching. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 19784–19795. Curran Associates, Inc.

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

Maziarz, K., Tripp, A., Liu, G., Stanley, M., Xie, S., Gaiński, P., Seidl, P., and Segler, M. (2023). Re-evaluating retrosynthesis algorithms with syntheseus. In *NeurIPS 2023 AI for Science Workshop*.

Mendes, B. B., Conniot, J., Avital, A., Yao, D., Jiang, X., Zhou, X., Sharf-Pauker, N., Xiao, Y., Adir, O., Liang, H., Shi, J., Schroeder, A., and Conde, J. (2022). Nanodelivery of nucleic acids. 2(1):24.

Mennen, S. M., Alhambra, C., Allen, C. L., Barberis, M., Berritt, S., Brandt, T. A., Campbell, A. D., Castañón, J., Cherney, A. H., Christensen, M., Damon, D. B., Eugenio de Diego, J., García-Cerrada, S., García-Losada, P., Haro, R., Janey, J., Leitch, D. C., Li, L., Liu, F., Lobben, P. C., MacMillan, D. W. C., Magano, J., McInturff, E., Monfette, S., Post, R. J., Schultz, D., Sitter, B. J., Stevens, J. M., Strambeanu, I. I., Twilton, J., Wang, K., and Zajac, M. A. (2019). The evolution of high-throughput experimentation in pharmaceutical development and perspectives on the future. 23(6):1213–1242. Publisher: American Chemical Society.

Miao, L., Li, L., Huang, Y., Delcassian, D., Chahal, J., Han, J., Shi, Y., Sadtler, K., Gao, W., Lin, J., Doloff, J. C., Langer, R., and Anderson, D. G. (2019). Delivery of mRNA vaccines with heterocyclic lipids increases anti-tumor efficacy by STING-mediated immune cell activation. 37(10):1174–1185.

Mitchell, M. J., Billingsley, M. M., Haley, R. M., Wechsler, M. E., Peppas, N. A., and Langer, R. (2021). Engineering precision nanoparticles for drug delivery. 20(2):101–124.

Nasreen, S., Chung, H., He, S., Brown, K. A., Gubbay, J. B., Buchan, S. A., Fell, D. B., Austin, P. C., Schwartz, K. L., Sundaram, M. E., Calzavara, A., Chen, B., Tadrous, M., Wilson, K., Wilson, S. E., Kwong, J. C., and on behalf of the Canadian Immunization Research Network (CIRN) Provincial Collaborative Network (PCN) Investigators (2022). Effectiveness of COVID-19 vaccines against symptomatic SARS-CoV-2 infection and severe outcomes with variants of concern in ontario. 7(3):379–385.

Pan, X., Wang, H., Li, C., Zhang, J. Z. H., and Ji, C. (2021). MolGpka: A web server for small molecule pka prediction using a graph-convolutional neural network. *Journal of chemical information and modeling*.

Patrignani, A., Schicchi, N., Calcagnoli, F., Falchetti, E., Ciampani, N., Argalia, G., and Mariani, A. (2021). Acute myocarditis following comirnaty vaccination in a healthy man with previous sars-cov-2 infection. *Radiology Case Reports*, 16(11):3321–3325.

Pedawi, A., Gniewek, P., Chang, C., Anderson, B. M., and van den Bedem, H. (2022). An efficient graph generative model for navigating ultra-large combinatorial synthesis libraries.

Petrucci, R. H., Harwood, W. S., and Herring, F. G. (2002). *General Chemistry: Principles and Modern Applications*. Prentice Hall, Upper Saddle River, NJ, 8th edition. 1 volume (various pagings): illustrations (some color); 26 cm.

Qin, S., Tang, X., Chen, Y., Chen, K., Fan, N., Xiao, W., Zheng, Q., Li, G., Teng, Y., Wu, M., and Song, X. (2022). mRNA-based therapeutics: powerful and versatile tools to combat diseases. 7(1):166.

RDKit (n.d.). RDKit: Open-source cheminformatics. Available online at https://www.rdkit.org. Accessed on 2024-06-05.

Ropp, P., Kaminsky, J., Yablonski, S., and Durrant, J. (2019). Dimorphite-dl: An open-source program for enumerating the ionization states of drug-like small molecules. *Journal of Cheminformatics*, 11.

Rüger, J., Ioannou, S., Castanotto, D., and Stein, C. A. (2020). Oligonucleotides to the (gene) rescue: Fda approvals 2017–2019. *Trends in Pharmacological Sciences*, 41(1):27–41.

Sacha, M., Blaz, M., Byrski, P., Wlodarczyk-Pruszynski, P., and Jastrzebski, S. (2020). Molecule edit graph attention network: Modeling chemical reactions as sequences of graph edits. *CoRR*, abs/2006.15426.

Sahin, U., Karikó, K., and Türeci, (2014). mRNA-based therapeutics — developing a new class of drugs. 13(10):759–780.

Schneider, N., Lowe, D., Sayle, R., and Landrum, G. (2015). Development of a novel fingerprint for chemical reactions and its application to large-scale reaction classification and similarity. *Journal of chemical information and modeling*, 55.

Schneider, N., Stiefl, N., and Landrum, G. A. (2016). What's what: The (nearly) definitive guide to reaction role assignment. 56(12):2336–2346. Publisher: American Chemical Society.

Schwaller, P., Laino, T., Gaudin, T., Bolgar, P., Hunter, C., Bekas, C., and Lee, A. (2019). Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS Central Science*, 5.

Segler, M. H. S., Preuß, M., and Waller, M. P. (2017). Towards "alphachem": Chemical synthesis planning with tree search and deep neural network policies. *CoRR*, abs/1702.00020.

Segler, M. H. S., Preuss, M., and Waller, M. P. (2018). Planning chemical syntheses with deep neural networks and symbolic AI. 555(7698):604–610.

Shi, C., Luo, S., Xu, M., and Tang, J. (2021). Learning gradient fields for molecular conformation generation.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., and Hassabis, D. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359.

Simm, G. N. C., Pinsler, R., and Hernández-Lobato, J. M. (2020). Reinforcement learning for molecular design guided by quantum mechanics.

Simonovsky, M. and Komodakis, N. (2018). Graphvae: Towards generation of small graphs using variational autoencoders.

Stern, S. T. and McNeil, S. E. (2007). Nanotechnology safety concerns revisited. 101(1):4–21. _eprint: https://academic.oup.com/toxsci/article-pdf/101/1/4/10976607/kfm169.pdf.

Sud, M., Fahy, E., Cotter, D., Brown, H. A., Dennis, E. A., Glass, C. K., Merrill, A. H. J., Murphy, R. C., Raetz, C. R. H., Russell, D. W., and Subramaniam, S. (2007). LMSD: Lipid maps® structure database. Nucleic Acids Research. PMID: 17098933.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Swanson, K., Liu, G., Catacutan, D. B., Arnold, A., Zou, J., and Stokes, J. M. (2024). Generative ai for designing and validating easily synthesizable and structurally novel antibiotics. *Nature Machine Intelligence*, 6:338–353.

van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks.

Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., and Fergus, R. (2013). Regularization of neural networks using dropconnect. In Dasgupta, S. and McAllester, D., editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1058–1066, Atlanta, Georgia, USA. PMLR.

Weininger, D. (1988). SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. 28(1):31–36. Publisher: American Chemical Society.

Wittrup, A., Ai, A., Liu, X., Hamar, P., Trifonova, R., Charisse, K., Manoharan, M., Kirchhausen, T., and Lieberman, J. (2015). Visualizing lipid-formulated siRNA release from endosomes and target gene knockdown. 33(8):870–876.

Xu, E., Saltzman, W. M., and Piotrowski-Daspit, A. S. (2021). Escaping the endosome: assessing cellular trafficking mechanisms of non-viral vehicles. *Journal of Controlled Release*, 335:465–480.

Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., and Tang, J. (2022). Geodiff: a geometric diffusion model for molecular conformation generation.

Xu, Y., Ma, S., Cui, H., Chen, J., Xu, S., Gong, F., Golubovic, A., Zhou, M., Wang, K. C., Varley, A., Lu, R. X. Z., Wang, B., and Li, B. (2024). AGILE platform: a deep learning powered approach to accelerate LNP development for mRNA delivery. 15(1):6305.

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., Palmer, A., Settels, V., Jaakkola, T., Jensen, K., and Barzilay, R. (2019). Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, 59(8):3370–3388. Correction published: J Chem Inf Model. 2019 Dec 23;59(12):5304-5305. doi: 10.1021/acs.jcim.9b01076.

You, J., Liu, B., Ying, Z., Pande, V., and Leskovec, J. (2018). Graph convolutional policy network for goal-directed molecular graph generation. In *Advances in Neural Information Processing Systems*, volume 31.

Yu, H., Kühne, R., Ebert, R.-U., and Schüürmann, G. (2010). Comparative analysis of qsar models for predicting pk(a) of organic oxygen acids and nitrogen bases from molecular structure. *Journal of chemical information and modeling*, 50:1949–60.

Zang, C. and Wang, F. (2020). Moflow: An invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '20. ACM.

Zhong, S. and Guan, X. (2023). Count-based morgan fingerprint: A more efficient and interpretable molecular representation in developing machine learning-based predictive regression models for water contaminants' activities and properties. 57(46):18193–18202. Publisher: American Chemical Society.

Appendix A

Algorithms

A.1 Algorithm of Guided Monte Carlo Tree Search for Lipid Generation

We define a `Node` class to represent the nodes appeared in the tree search. The `Node` class has the following attributes:

- `state`: Stores the SMILES representation of the molecule represented by this node.
- `N`: A visit count of the node, initialized to zero.
- `P`: The prior probability assigned based on predictions from a policy network.
- `W`: The cumulative value sum, representing the total assessed value of this node's state.
- `children`: A dictionary to hold child nodes, with the keys of the dictionary to be actions (i.e., the next building blcoks to select) and the values to be corresponding child nodes.

Algorithm 1 Policy Network Guided MCTS

Require: Product(): reaction predictor
Require: PropertyScore(): property score predictor
Require: SearchProbability(): search probability calculator

$f_\theta \leftarrow$ randomly initialized neural network with parameter θ
$\lambda \leftarrow$ regularization constant
$\alpha \leftarrow$ learning rate
$c \leftarrow$ exploration weight
$D = \emptyset$

for each iteration do
    for each play do
        $S_0 \leftarrow$ empty root state
        $D_{play} \leftarrow \text{MCTS}(S_0)$
        $D = D \cup D_{play}$
    end for
    for each epoch do
        Train f_θ using D
    end for
    Reset $D = \emptyset$
end for

function MCTS(*root_state*)
    Initialize *root_node* with *root_state*
    EXPAND(*root_node*)
    generation $= \emptyset$
    for each simulation do
        $leaf\_node, search\_path \leftarrow \text{SELECT}(root\_node)$
        if *leaf_node* is two-tail lipid **then**
            v = PropertyScore(*leaf_node*)
            Add *search_path* to *generation*
        else
            EXPAND(*leaf_node*)
            $v \leftarrow \text{ROLLOUT}(leaf\_node)$
        end if
        BACKPROPAGATE(v, *search_path*)
    end for
    Write *generation* to log file
    return visit counts of all state-action pairs
end function

Algorithm 2 Select Function in MCTS

function SELECT(*root_node*)
    $search\_path = [\,]$
    $node \leftarrow root\_node$
    while $node.children$ is not empty **do**
        $selected\_action = \arg\max_a \text{UCB\_SCORE}(node, node.children[a])$
        $node \leftarrow node.children[selected\_action]$
        Add *node* to *search_path*
    end while
    Update $node.state$ with Product($node.state, selected\_action$)
    return *node*, *search_path*
end function

Algorithm 3 Expand Function in MCTS

function EXPAND(*node*)
    $A(node) \leftarrow \text{NEXT\_BUILDING\_BLOCKS}(node)$
    $P(node, \cdot) = f_\theta(node.state, A(node))$
    for a in $A(node)$ **do**
        Initialize *child_node* with $P(node, a)$
        $node.children[a] = child\_node$
    end for
end function

Algorithm 4 Backpropagate Function in MCTS

function BACKPROPAGATE(v, *search_path*)
    for *node* in *search_path* **do**
        $node.N \leftarrow node.N + 1$
        $node.W \leftarrow node.W + v$
        $root\_node.N \leftarrow root\_node.N + 1$
    end for
end function

Algorithm 5 Rollout Function in MCTS

function ROLLOUT(*node*)
    while $node.state$ is not two-tail lipid **do**
        $A(node) \leftarrow \text{NEXT\_BUILDING\_BLOCKS}(node)$
        return 0 if $A(node)$ is empty
        $a \leftarrow$ a random choice from $A(node)$
        $node \leftarrow$ new node with Product($node.state$, a)
    end while
    return PropertyScore(*node*)
end function

Algorithm 6 UCB Score Calculation Function in MCTS

function UCB_SCORE(*parent_node*, *child_node*)
    if $child\_node.N = 0$ **then**
        $Q = 0$
    else
        $Q = \frac{child\_node.W}{child\_node.N}$
    end if
    $U = c \cdot child\_node.P \cdot \frac{\sqrt{parent\_node.N}}{child\_node.N+1}$
    return $Q + U$
end function

Algorithm 7 Get Action Space Function in MCTS

Require: *max_expand_num* a pre-determined number
function NEXT_BUILDING_BLOCKS(*node*)
    if *node* is empty node **then**
        $A(node) \leftarrow$ *max_expand_num* random samples from head search space
    else
        $reactive\_tail\_set \leftarrow$ tails which can react with $node.state$
        $A(node) \leftarrow$ *max_expand_num* random samples from $reactive\_tail\_set$
    end if
    return $A(node)$
end function

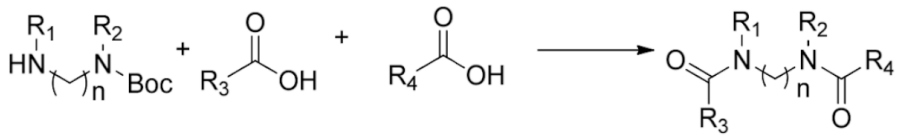
Appendix B

Template-based Reaction Prediction

B.1 Reaction Templates

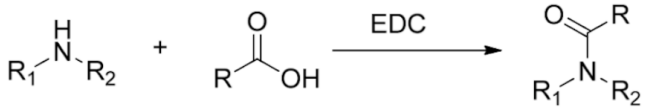
We utilize 13 reaction templates sourced from (Grygorenko et al., 2020), as outlined below. Within our generative model framework, we focus exclusively on reactions involving two reactants, which leads us to exclude Reaction 1 from our simulations. Additionally, Reactions 6, 7, 9, and 12 all involve a linker molecule characterized by a trifluoroethyl substructure. It is important to note that this linker molecule is not explicitly decoded into the reactions in our in silico modeling; however, specific conditions are required for its use in laboratory experiments. Furthermore, any catalysts present in these reactions are also not incorporated into our in silico coding, reflecting a simplification to facilitate computational modeling while acknowledging the complexities that arise during actual chemical synthesis.

Reaction 1



Acylated-acylated diamine: diamine (mono Boc-protected) + acid + acid; one-pot Boc cleavage

Reaction 2

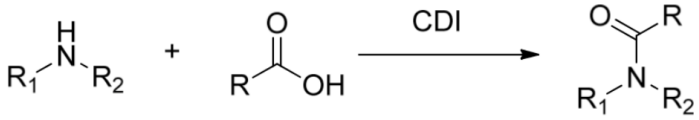


Amide: amine + acid + EDC

Reaction 3

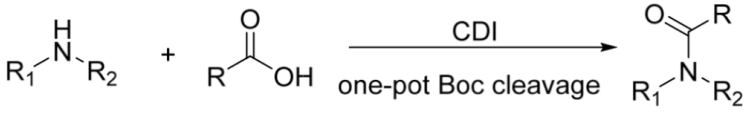
Amide: amine + acid + CMPI

Reaction 4



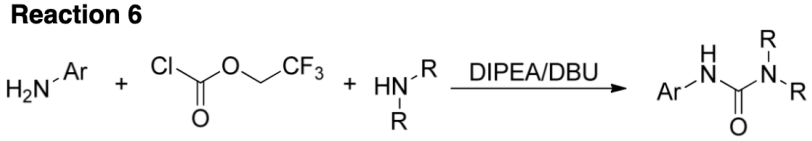
Amide: amine + acid + CDI

Reaction 5



Amide: amine (contains Boc-protected amino group in side chain) + acid + CDI; one-pot Boc cleavage

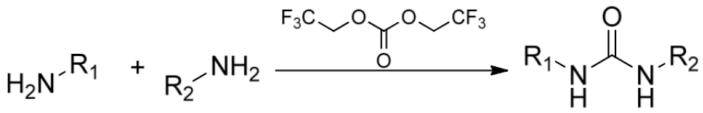
Reaction 6



R = Alk, H

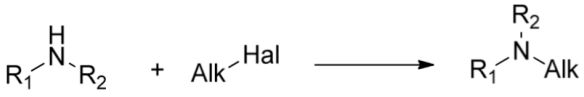
Urea: amine + amine + 2',2',2'-trifluoroethyl chlorocarbonate + DIPEA

Reaction 7



Urea: amine + amine + bis(2,2,2-trifluoroethyl)carbonate

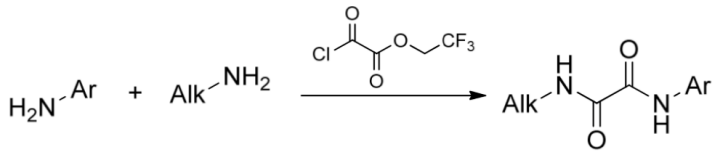
Reaction 8



R1, R2: Alk, Ar, cycle

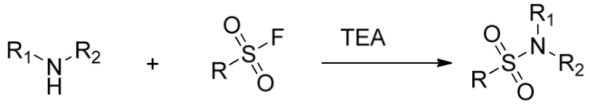
Alkyl amine: secondary amine + alkyl halide

Reaction 9



Oxamide: aryl amine + primary/secondary alkyl amine + 2',2',2'-trifluoroethylchloroxoformiate

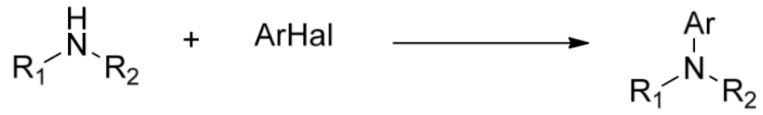
Reaction 10



R1, R2: Alk, H

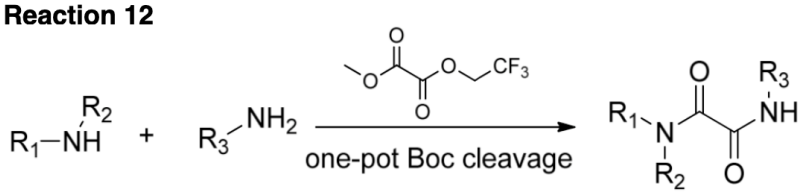
Sulfonamide: amine + sulfonyl chloride + TEA

Reaction 11



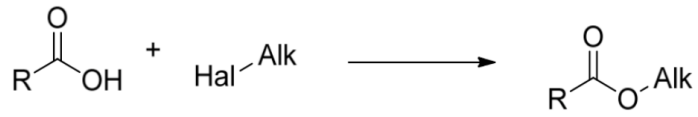
Alkyl aryl amine: alkyl amine + aryl halide

Reaction 12



Oxamide: amine + secondary alkyl amine + methyl (2,2,2-trifluoroethyl) oxalate

Reaction 13



Ester: acid + alkyl halide

Appendix C

Example Synthesis Planning

C.1 Selected Examples of Synthesis Planning

Figures C.1 and C.2 give two examples of our generated ionizable lipids with their corresponding synthesis paths. Note that these synthesis paths have been validated by the retrosynthesis tool Syntheseus (Maziarz et al., 2023). In the figures below, the blue nodes represent product molecules (i.e., intermediate product or final product) and the green nodes represent building block molecules (i.e., lipid head building block or lipid tail building block).

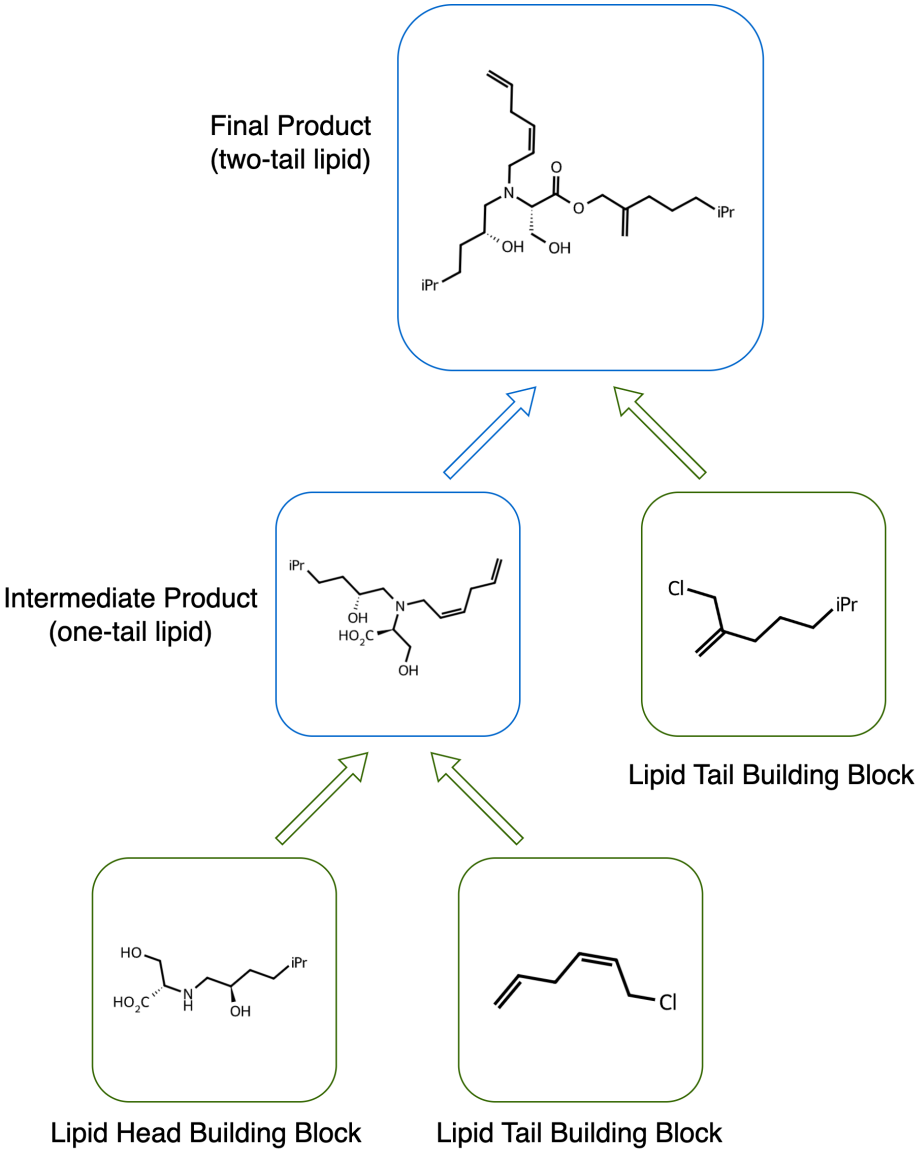


Fig. C.1 Example 1 of generated ionizable lipid with synthesis path.

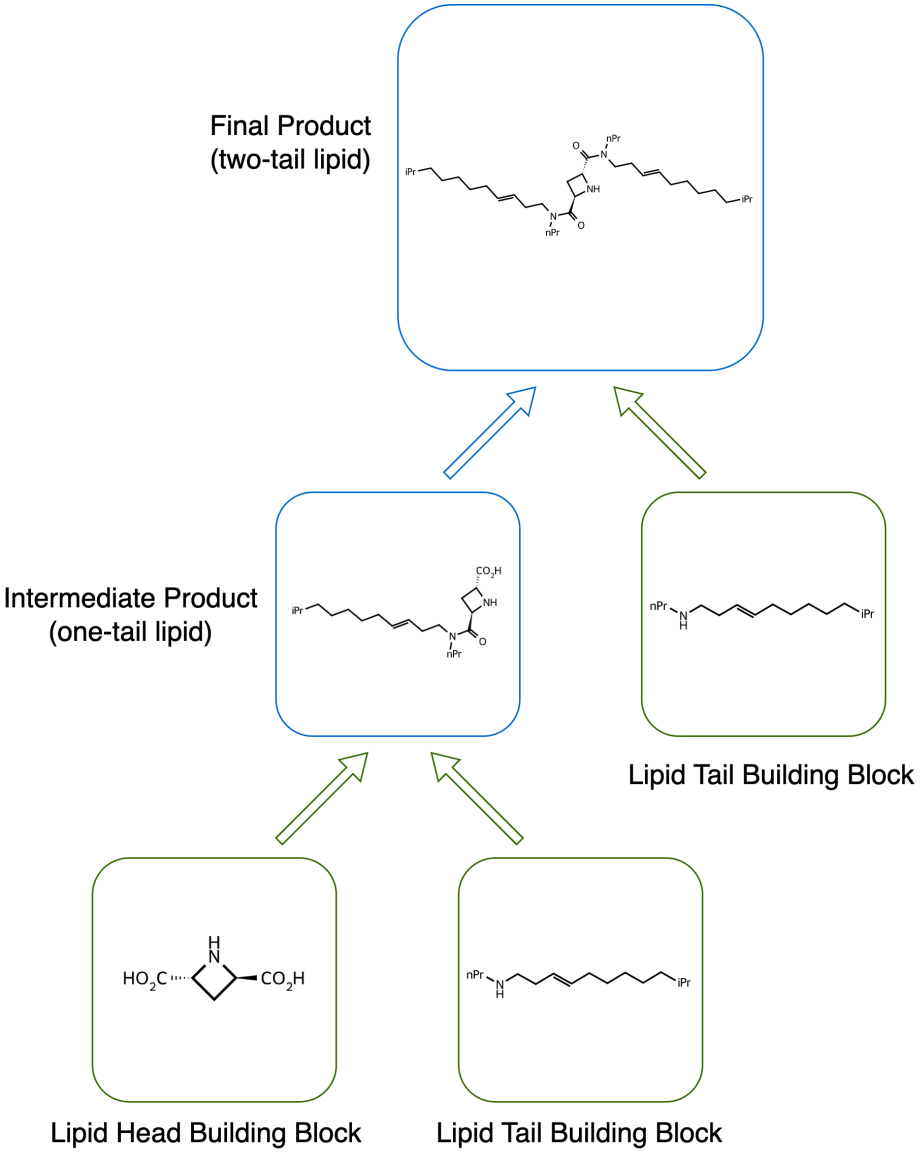


Fig. C.2 Example 2 of generated ionizable lipid with synthesis path.