Machine Learning for Inverse Problems in Chemistry: Spectra to Structure

Thesis submitted in partial fulfillment
of the requirements for the degree of

*(Master of Science in **Computational Natural Sciences** by Research)*

by

S Bhuvanesh
2018113002
bhuvanesh.sridharan@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Machine Learning for Inverse Problems in Chemistry: Spectra to Structure" by S Bhuvanesh, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Advisor: Prof. U Deva Priyakumar

To the joy of problem solving.

Acknowledgments

I would like to express my sincere gratitude to Dr. U Deva Priyakumar, for his unwavering support and guidance throughout the duration of my work. The invaluable experience gained under his supervision would go a long way in shaping my future career.

I would also like to extend my heartfelt thanks to Sarvesh Mehta for his invaluable contributions and insightful suggestions, which have proven crucial in the completion of the core work of my thesis. Additionally, I would like to thank all my lab members who have supported me with their constant feedback and assistance. In particular, I would like to thank Yashaswi Pathak for helping me learn foundational concepts at the beginning of my work, and Manan Goel for his collaboration on a project important for my thesis. I would like to extend a heartfelt acknowledgment to my juniors, with whom I had the pleasure of engaging in stimulating discussions on a range of research topics. I also owe a lot to Animesh, Akshit and the whole CH gang of friends for being there for me whenever I needed them.

Lastly, I would like to express my gratitude to my parents, brother, manni, and niece Shrinu for their unconditional love and unwavering support during my academic journey. This accomplishment would not have been possible without their love and sacrifices.

Abstract

The discovery of new molecules and materials helps expand the horizons of novel and innovative real-life applications. In the pursuit of finding molecules with desired properties, chemists have traditionally relied on experimentation and recently on combinatorial methods to generate new substances often complimented by computational methods. The sheer size of the chemical space makes it infeasible to search through all possible molecules exhaustively. This calls for fast and efficient methods to navigate the chemical space to find substances with desired properties. This class of problems is referred to as inverse design problems. There is a variety of inverse problems in chemistry encompassing various subfields like drug discovery, retrosynthesis, structure identification etc. Recent developments in modern machine learning (ML) methods have shown great promise in being able to tackle problems of this kind. This has helped in making major strides in all key phases of molecule discovery ranging from *in silico* candidate generation to their synthesis with focus on small organic molecules. Optimization techniques like Bayesian optimization, reinforcement learning, attention-based transformers, deep generative models like variational autoencoders and generative adversarial networks form a robust arsenal of methods. The first chapter of this thesis summarizes the development of deep learning to tackle a wide variety of inverse design problems in chemistry towards the quest for synthesizing small organic compounds with purpose.

Spectroscopy is the study of how matter interacts with electromagnetic radiations of specific frequencies that has led to several monumental discoveries in science. The spectra of any particular molecule is highly information-rich; while structure to spectra is straightforward using computational methods, the inverse relation of spectra to the corresponding molecular structure is still an unsolved problem. Nuclear Magnetic Resonance (NMR) spectroscopy is one such critical technique in the scientists' toolkit to characterise small organic molecules to biomolecular structures like proteins and nucleic acids. In the second half of the thesis, a novel machine learning framework is proposed that attempts to solve this inverse problem by navigating the chemical space to find the correct structure given an NMR spectra. The proposed framework uses a combination of online Monte-Carlo-Tree-Search (MCTS) and a set of offline trained Graph Convolution Networks to build a molecule iteratively from scratch. Our method is able to predict the correct structure of the molecule $\sim 80\%$ of the time in its top 3 guesses. We believe that the proposed framework is a significant step in solving the inverse design problem of NMR spectra to molecule that would be a significant step forward in high-throughput molecular synthesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

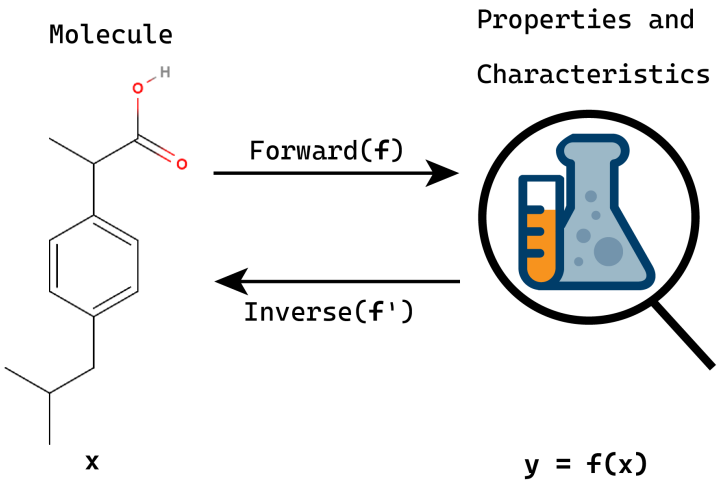


Figure 1.1: Forward problems are those where in we evaluate properties for a molecule x whereas inverse involve find about the molecule x given the observed properties y

Historically, chemical advancements are driven by experimentation and synthesis of new compounds, followed by evaluation of their properties and quirks. The intent being the discovery of novel compounds for novel applications and uses. Understanding of the structure-property relationships plays a major role in this process. Traditional computational chemistry methods in addition to experiments have been shown to be invaluable. Methodological improvements are imperative to keep up with the need of novel molecules that exhibit required properties as time progressed. For instance, the average time for a drug to reach the market is about 13 years.[1] Fortunately, the vast improvement in computational capacity in tandem with advancements in artificial intelligence and algorithms has enabled chemists to approach this problem from a different dimension. One such way is to first look at an application with certain desired requirements and attempt to design substances directly while keeping the required properties

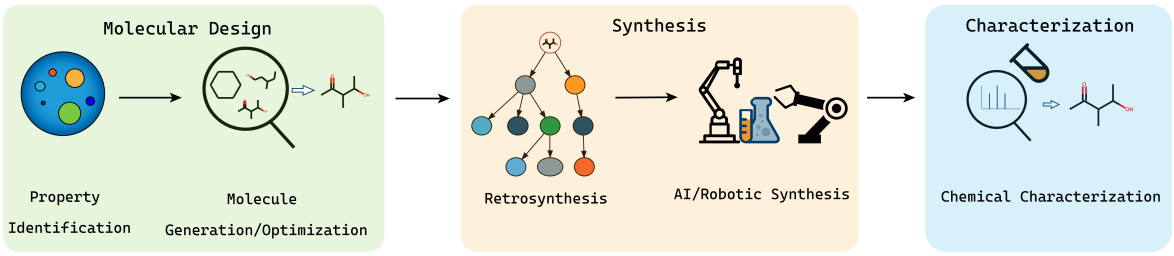


Figure 1.2: Pipeline of molecular design to realization

and characteristics in mind. Thus, the problem of molecule discovery has been modelled as an inverse problem.

Machine Learning(ML) advancements in the recent years has enabled us to approach molecule discovery from such new dimension. ML as a problem-solving paradigm has many important applications across various fields. This ML boom has been fuelled by both the increase in computational capacity and the increase in the amount of data available to train the frameworks. There have been various efforts in expanding the set of available libraries of molecules, and their properties.[2, 3, 4, 5, 6, 7] Contrary to a traditional knowledge engineering approach where the programmer provides the explicit algorithm to process the input, ML algorithms try to fit a function to the given data while also generalizing the pattern. Hence, ML approaches are an effort to enable a machine to "learn" the underlying science from examples(dataset). Review articles by Strieth-Kalthoff et al. [8] and Butler et al. [9] give an introduction to ML from the perspective of synthetic chemistry and also highlight how ML has advanced research in chemical sciences.

Inverse problem refers to the class of problems wherein the task is to deduce or evaluate the set of causal factors that led to a particular set of observation or measurements (Figure 1.1).[10, 11] Inverse problems are of great interest in various fields of science owing to the fact that they reveal a lot about the underlying relations which are not directly observed.[12, 13, 14, 15] Many of the inverse tasks pertaining to chemistry belong to the subclass of non-linear inverse problems which are complex to deal with.[16, 17] In such problems, the forward function $y = f(x)$ is a non-linear relation between the input x and the output y.

In essence, the need is to do the following tasks in order: discover new molecules, simulate or evaluate their potential suitability for a task, find methods to synthesize those molecules, followed by characterizing the molecules generated(Figure 3.2). The final goal would be to be able to do the above tasks as a seamless process which is otherwise arduous and time consuming. This effective "closing of the loop" would lead to an ideal pipeline which would propel the discovery/validation/realization of novel molecules for novel applications.

In the next chapter, various inverse problems that are relevant to the process of molecule design are discussed. Once the requirement of the properties are finalized, the first task is to identify molecules and their structure which would exhibit desired properties, which is termed as molecule generation. Once

the structure of a target molecule is known, the next task is to find a viable reaction pathway using a set of available precursor molecules to synthesize the target molecule, which is the task of retrosynthesis. Once both the target molecule, and its synthesis logic is attained, the next step to automate is the actual synthesis of the substance using AI assisted robots. Even though this is not exactly an inverse problem, we are discussing it briefly in this article as it fits in the overarching task of leading a molecule from design to realization. Once we have realized a sample of a substance in the lab, it is important to ensure that the substance synthesized is actually the one which was intended, which is the task of chemical characterization. The second half of the thesis approaches one such tasks of chemical characterization using the paradigm of deep reinforcement learning.

The following chapter describes some of the important neural architectures that are commonly used in the works attempting to solve the inverse problems. The chapter also discusses recent ML based advancements in each of the four subtasks that we enumerated above.

Chapter 2

Inverse Problems in Chemistry: Molecular Design to Realization

2.1 Brief Overview of Modern ML Methods used for Inverse Problems

This section gives a brief overview of some of the commonly used modern ML methods which are essential to understand the recent work in the domain of inverse problems of molecular design.

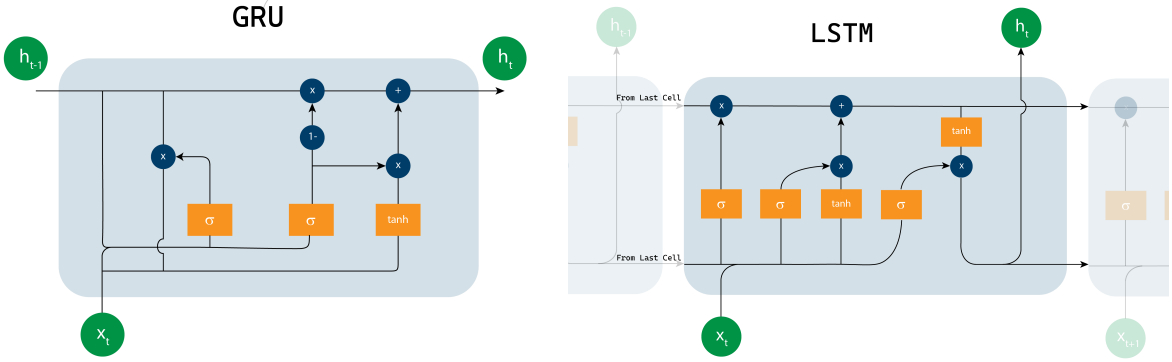


Figure 2.1: Architecture of (a) Gated Recurrent Unit's cell and (b) Long-Short-Term-Memory's cell. x_t, h_t and c_t are the input token, hidden state and cell state respectively and σ represents the sigmoid activation function. $\times$ and $+$ represent elementwise addition and product. The input tokens and hidden states are passed through these cells recursively to get the final output.

2.1.1 Recurrent Neural Networks

Recurrent neural networks generalise feed forward neural networks to be able to handle sequential data. They can remember what was previously seen in the input and help provide context for elements that occur later in the sequence. The SMILES representation of molecules is formed by strings and is hence, sequential in nature. A RNN generally consists of what is known as a hidden state which can be interpreted as a memory unit which remembers what occurred in the sequence. Every token in the SMILES string is converted to machine readable vector which is combined with the hidden state to

provide a new hidden state.[18] At time t, The general update rule for an RNN is given by

$$h_t = f(x_t, h_{t-1}) \tag{2.1}$$

where x_t is the input token and h_{t-1} is the hidden state after the previous input.

RNNs are optimized using an algorithm called back propagation through time.[19] During back propagation, each gradient is calculated with respect to the effects of the gradient in the next step. However, this also brings a problem: if the magnitude of the gradient at the previous step is small, then the magnitude at the current step is even smaller which means that the effect of the initial tokens does not reach the final calculated gradient. This is called the vanishing gradient problem. In order to tackle these, two specialized RNN architectures have been developed (Figure 2.1) -

- Long Short Term Memory (LSTM): In the LSTM architecture,[20] another state is added along with the hidden state called cell state. It can be thought of as a memory unit which contains relative information way down the sequence chain and since, it retains information from earlier steps, the information from earlier steps is available in the later steps. The information to be retained and forgotten is controlled using three gates which use the hidden state, cell state and input at the current step to calculate the hidden state and cell state for the next step.

- Gated Recurrent Unit (GRU): The GRU[21] architecture is similar to LSTM but instead of three, they use only two gates and do not contain a cell state. Only the hidden state is used to carry information. Due to the fewer gates, the number of operations is lesser in GRUs in comparison to LSTMs and are hence, slightly faster but show similar accuracy.

RNNs can be used to generate text by using the hidden state at the current step to forecast the token that is most likely to appear at the next step and add it to the text generated so far and repeat unless a token is generated which signifies the end or a maximum specified length is reached. This can be applied to generate SMILES strings but it posts a problem that the resultant string may not represent a molecule. Several ML architectures have been proposed to generate valid molecules which have RNNs as the core of their generator.[22, 23, 24, 25, 26]

2.1.2 Graph Neural Networks

The graph representation of a molecules opens up the avenue for a wide variety of algorithms that can be used directly on the graph structure. Each atom is represented by a node in a graph and each bond by an edge. These vertices and edges are differentiated from each other by the presence of a feature vector corresponding to each vertex and edge. The atom feature x_v for an atom v may consist of information like one hot encoding of atom type, hybridization, formal charge etc. Similary, the bond feature e_{uv} for a bond between atoms u and v contains information like bond type and stereochemistry etc.

Most graph neural networks are different variants of a common architecture. This architecture consists of two phases -

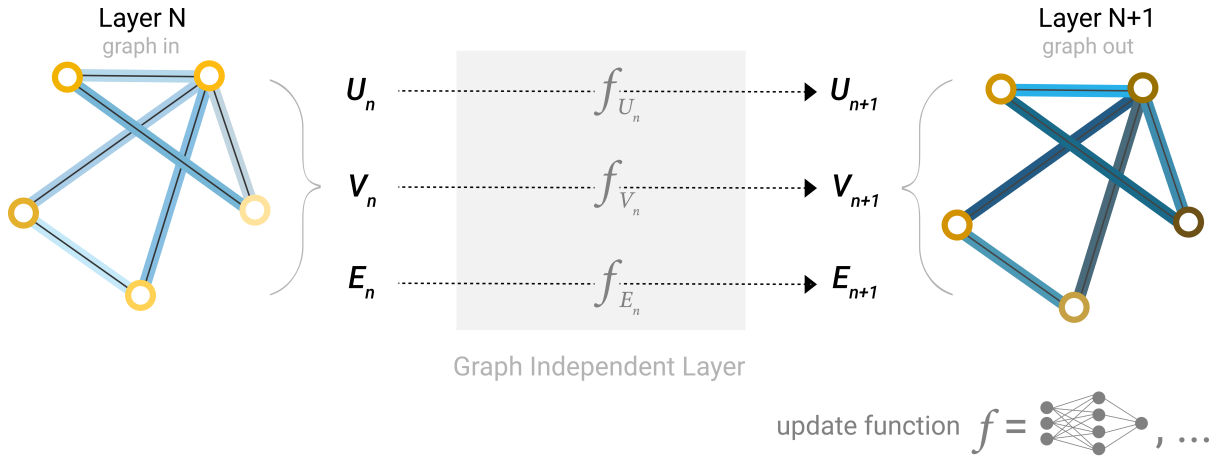


Figure 2.2: Overview of Graph Neural Networks. A single layer of a simple GNN. A graph is the input, and each component (V,E,U) gets updated by a MLP to produce a new graph. Each function subscript indicates a separate function for a different graph attribute at the n-th layer of a GNN model. Figure from Sanchez-Lengeling et al. [27] under Creative Commons licence

- Message Passing: The message passing phase is responsible for capturing the environmental information around a node. This phase is run for T timesteps and at the ith timestep, information from all nodes that are i edges away reaches the respective node. It is defined in terms of the message function M_t and vertex update function U_t. At every timestep t, each node has a hidden state h_v^t and $h_v^0 = x_v$ and are updated using a message vector m_v^{t+1} according to

$$\begin{aligned} m_v^{t+1} &= \sum_{u \in N(v)} M_t(h_v^t, h_v^t, e_{uv}) \\ h_v^{t+1} &= U_t(h_v^t, m_v^{t+1}) \end{aligned} \tag{2.2}$$

  where $N(v)$ is the set of neighbours of v.

- Readout: In this phase, a feature vector for the entire graph is calculated using some differentiable readout function R

$$\hat{y} = R(\{h_v^T | v \in G\}) \tag{2.3}$$

  where G is the set of vertices in the graph.

A general overview of a single layer in a simple graph neural network is given in Figure 2.2.

Different graph neural networks use different functions for message passing and readout which can be used for predicting molecular properties or constructing the graph by adding a node at every timestep taking into account the graph constructed till that timestep. Building molecules in the form of graphs brings an advantage that unlike intermediate states of SMILES strings being invalid, it is much easier to make sure that each constructed subgraph is always valid. GNNs have proven to be a great tool to featurize the molecules and hence the featurized vectors can be used for further downstream prediction

tasks.[28, 29] Such featurization of the current state of the molecule can also help in driving feedback to other parts of an architecture to guide the design of molecules.[30, 31]

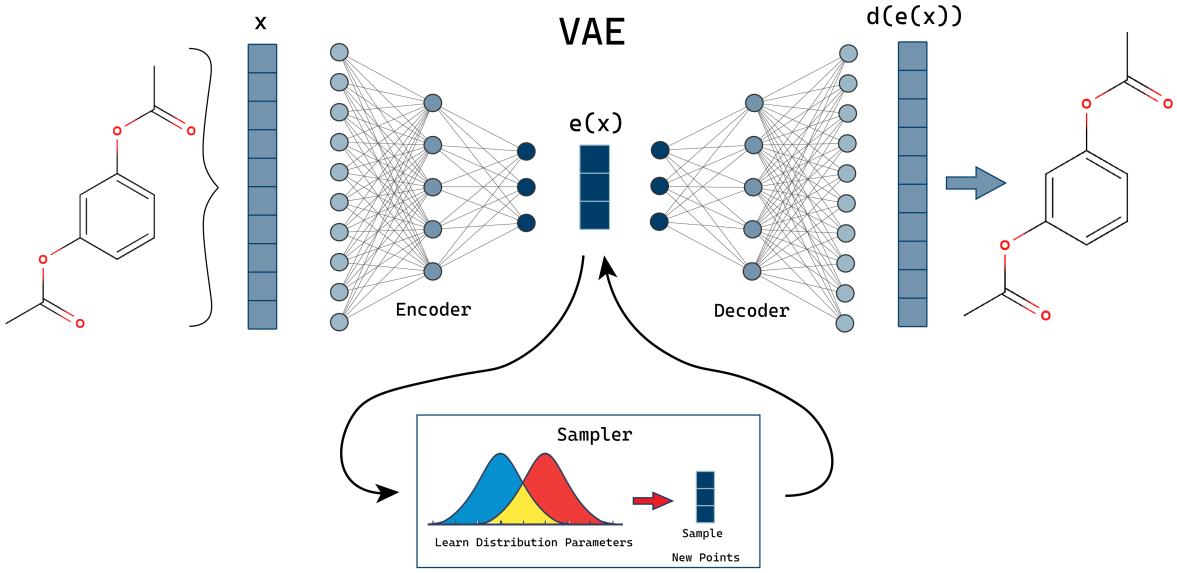


Figure 2.3: Overview of Variational Autoencoders. The input molecule (x) is encoded in a continuous latent space by estimating parameters of a normal distribution from which the observation is sampled $e(x)$. The decoder then tries to reconstruct the input from $e(x)$ such that $d(e(x))$ and x belong to an identical probability distribution.

2.1.3 Variational Autoencoders

Autoencoders are mainly designed to encode the input into a meaningful and compressed representation and then decode it back to get the initial input. In theory, an autoencoder would extract only the information from the input which is necessary to reconstruct the input from the smaller representation also known as the latent space representation. Mathematically, the problem is to find two functions $f_1 : \mathbb{R}^n \rightarrow \mathbb{R}^p$ and $f_2 : \mathbb{R}^p \rightarrow \mathbb{R}^n$ which satisfy -

$$argmin_{f_1,f_2}\mathbb{E}[\Delta(x, f_2 \cdot f_1(x))] \tag{2.4}$$

where Δ is the reconstuction loss.[32, 33]

Variational autoencoders are a variant of this architecture which provide a probabilistic manner for describing an observation in the latent space (Figure 2.3).[34] Instead of giving a single value for each latent space attribute like a conventional autoencoder, VAEs provide a probability distribution for each attribute. A latent space representation is then sampled from, the obtained probability distribution for each attribute from the encoder providing a continuous latent space representation. The probabilistic decoder can be assumed to be a generative model conditioned on a random latent variable z with parameters θ which gives a prior distribution on latent variables $p_\theta(z_i)$. Similarly, the encoder is equivalent

to an approximate posterior distribution over z given a datapoint x governed by parameters ϕ. The objective function is calculated using the marginal log-likelihood. The first term is the Kullback-Leibler divergence of the true posterior and the approximate prior. The second term is called the variation lower bound on the marginal likelihood and is defined as -

$$L(\theta, \phi; x_i) = -D_{KL}(q_\phi(z|x_i)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x_i)}[logp_\theta(x_i|z)] \tag{2.5}$$

Hence, the above mentioned objective function should be maximized for all data points with respect to θ and ϕ. A wide variety of models have been used for the encoder and the decoder including convolutional neural networks, graph convolution neural networks, RNNs and more. RNNs for SMILES strings and graph convolution for molecular graphs are the conventional encoders and decoders of choice in the domain of chemistry.[35, 36, 37, 38, 39, 40]

2.1.4 Generative Adversarial Networks

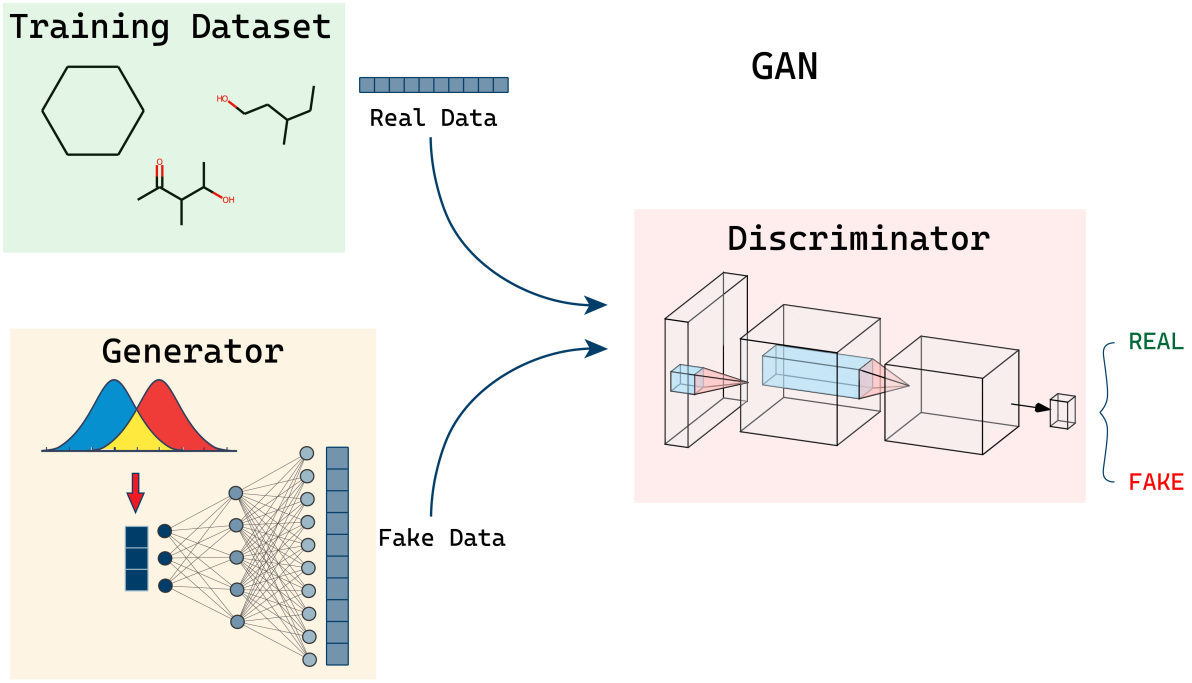


Figure 2.4: Overview of Generative Adversarial Networks. The generator generates fake data samples and and some samples are picked at random from the actual training set. These samples are then sent to the discriminator which classifies if the provided samples are real or fake. The generator and discriminator are then trained such that the generator tries to fool the discriminator and discriminator tries to correctly identify the fake samples.

GANs (Generative Adversarial Networks) are a set of models: a generator and discriminator. The two models are pitched against each other and trained (Figure 2.4). The generator attempts to capture

the distribution of a training dataset and create new sample data points similar to the training samples. The generator model is expected to do so without having direct access to the training samples but with feedback from the discriminator model. The discriminator is a classifier that is fed input from both the original training dataset also datapoints created by the generator. The role of the discriminator is to correctly discriminate and classify these points as either being generated by the generator or being a true datapoint. The process of training both these models is a mini-max problem wherein the discriminator wants to correctly distinguish all the samples and the generator wants the samples generated by it to be indistinguishable from the training distribution. This back-and-forth optimisation is said to be terminated when the models reach a saddle point which is minimum along one axis and maximum along another. The most impactful paper on GANs by Goodfellow et al. [41] trained their GAN models on the following objective function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] \ + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))] \tag{2.6}$$

Where $p_d(x)$ is the training data distribution and $p_z(z)$ is a predefined normal distribution from which the generator samples points. It is to be noted that the second term i.e. $\log(1 - D(G(z)))$ may tend to negative infinity when the training starts if the discriminator is stuck in a local minima. This is commonly referred to as mode collapse. On the contrary, if the discriminator function is trained too optimally, the model finds it difficult to train the generator since this leads to a weak gradient and hence slower training. Arjovsky et al. [42] found a way to overcome this issue by using an alternative method of training. Use of Wasserstein loss motivates the critic to maximise the distance of distribution of its output to real data and fake data. One of the key points of WGAN is the use of a linear layer instead of sigmoid layer for the output layer of the critic model. It is recognised in literature that WGANs provide better stability of training and reduces problems like mode collapse or vanishing gradients. Many of the papers discussed in the following sections like ORGAN,[43] MolCycleGAN,[44] employ WGANs in addition to GANs for their studies and evaluations.

2.1.5 Reinforcement Learning

Reinforcement learning is a class of machine learning algorithms which has gotten increasingly popular. They generally consist of two parts: an agent which performs actions and a critic which rewards or penalizes those actions (Figure 2.5). The system is described as a variable s_t which the agent parameterized by θ uses as input to predict an action a_t such that it leads to the maximum possible cumulative reward forming a markov decision process. The trajectory of the system is defined as $(s_0, a_0, s_1, a_1.....a_{T-1}, s_T)$ where s_T is called the terminal state and states between s_0 and s_T are called intermediate states. The total reward for the trajectory is calculated as a sum of rewards from the intermediate states and the terminal state.[45]

A popular idea in most reinforcement learning based algorithms is the Q function. The Q function takes the state s and an action a as input and returns the expected reward for the state action pair. If the strategy for choosing the actions is optimal, then at every state (s) the best action (a) will be taken

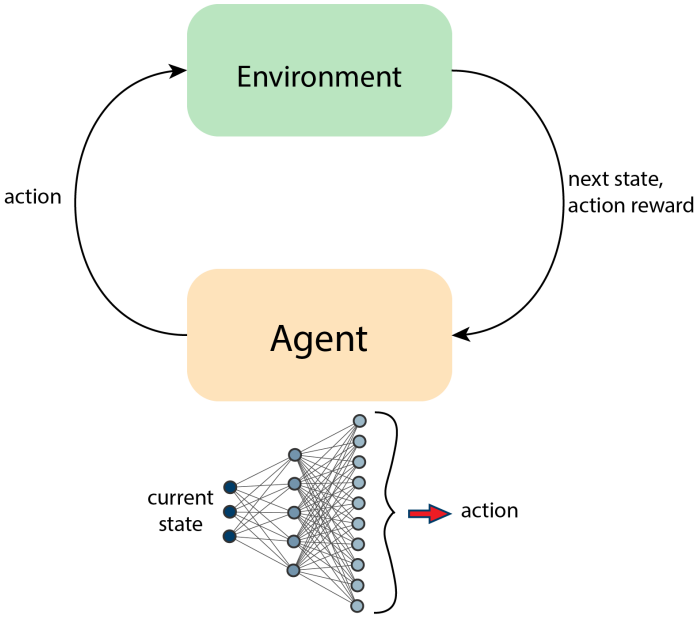


Figure 2.5: Overview of general reinforcement learning methods. An RL pipeline consists of two parts: the environment and the agent. The agent interacts with the environment by taking actions and these actions are then rewarded or penalized depending on if they lead to a more promising state. For example, in a game if the action takes the agent closer to victory, the agent is rewarded and if the action leads to a loss, the agent is then penalized.

which will lead to the best value of $Q(s, a)$. If the system is small with few states and few actions, we can ideally create a table which maps state action pairs to the respective Q values and this is is called Q learning. However, as the systems become larger, enumerating all possible state action pairs becomes infeasible. Hence, the Deep Q-Learning algorithm was proposed in which the instead of building a table, an artificial neural network is used to map input states to the (action, Q-value) pair. The best possible action is chosen with a probability ϵ and a random action is chosen with probability $1 - \epsilon$ to make sure the obtained information is exploited and new regions of the space are explored. This is called the Epsilon-Greedy Exploration Strategy. The values in the Q table for both cases are updated using the Bellman Equation where α and γ are the learning rate and discount factor respectively.

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(R_t + \gamma max_{a'}Q(s', a')) \tag{2.7}$$

Another popular method for reinforcement learning is maximizing the rewards using policy gradients. The expected reward can be calculated as a function of the parameters of the machine learning model (θ) -

$$J(\theta) = \mathbb{E}_\pi[r(\tau)] \tag{2.8}$$

where τ is the trajectory and π is the policy. The given objective function $J(\theta)$ can be maximized using gradient ascent. The term $r(\tau)$ in the equation above is approximated using a wide variety of algorithms like REINFORCE and Actor-Critic.

RL frameworks have proved themselves in the application of chemistry related tasks especially towards molecule optimization.[46, 31, 47, 22] They also have shown promise in tasks like reaction and geometry optimization.[48, 49, 50]

2.2 Inverse Problems in Molecule Discovery

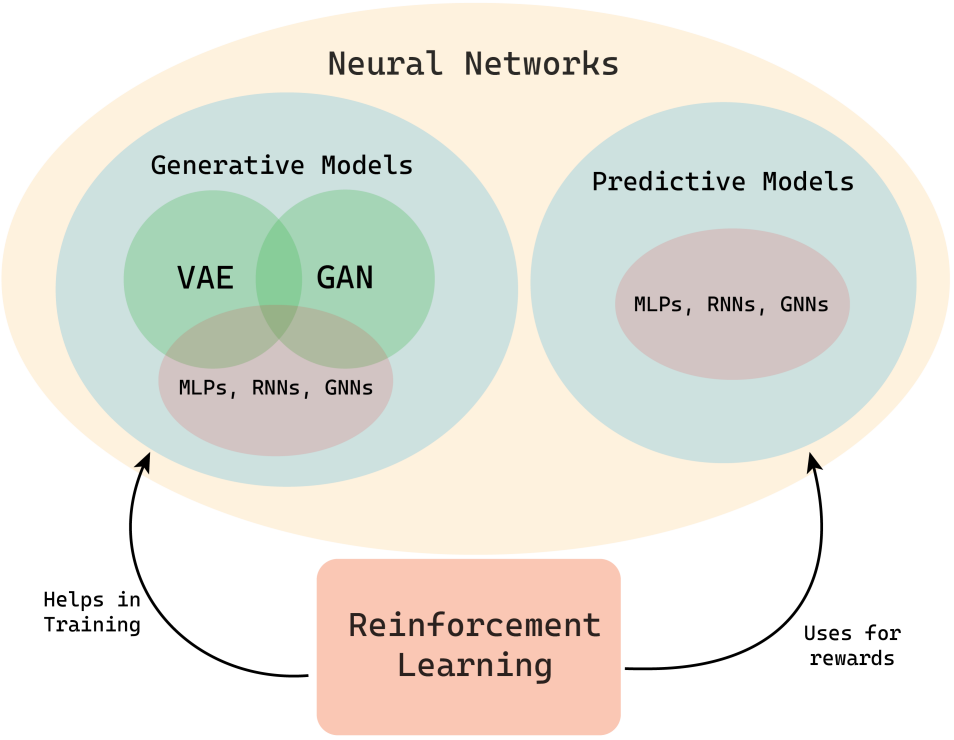


Figure 2.6: An overview of components and advancements in deep learning for the task of molecule generation: VAEs are an important framework that featurize molecules into an explorable latent space from which we can sample new points. GANs are useful to generate new data points from a sampled normal distribution. GANs are frameworks that use adversarial training to create points from a distribution similar to the training dataset. These generative models employ a variety of neural networks like RNNs and GNNs to do so. Further, QSAR deep predictive models can be used by various RL algorithms as a form of reward to aid the training of the generative models. These rewards can also be used as a means to train the models to optimize for properties of interest.

2.2.1 Molecule Generation

A critical step in the journey of finding novel molecules for desired applications is the process of molecule design. Finding appropriate candidates having a certain set of required properties or characteristics is an exceedingly difficult task in the chemical discovery pipeline. This is due to the enormous nature of the chemical space to be explored. The number of total drug-like molecules has been estimated

to be up to 10^{60}.[51] Candidate molecule generation can be modelled as an inverse problem wherein the intent is to find the optimum molecule and its structure that has a specific set of properties. Traditionally, computational means to find molecules of interest is to generate a large library of molecules through combinatorial methods.[52, 53, 54, 55]Then these large library of molecules are screened for desirable properties or experimental outcomes followed by optimization of the structure based on the understanding of the property-structure relationship. With advancements and progress in modern machine learning methods, there are reliable methods to accurately predict many properties for molecules at a rapid pace.[56]These ML methods try to capture the function that relates molecules to properties of interest. This has led to development of high throughput virtual screening(VS) methods that make it possible for us to narrow down possible candidates from a large library of molecules at a much faster rate than that possible with traditional methods.[57, 58]

In spite of these advancements, a significant computational effort is required to screen these huge libraries of molecules which may reach sizes beyond billion in number.[3, 4] This calls for methods that generate molecules in a more targeted way and explore chemical space more efficiently. *de novo* design of molecules contrast with the earlier discussed virtual screening method in a way the structure of the molecules are known *a priori*. Whereas in *de novo* design, the molecules are generated from scratch with optimization as the goal. The intent in *de novo* molecular design is to consider and evaluate lesser number of molecules than one would in screening.

One popular method of optimization is the class of variants of Genetic Algorithms.[59, 60] They involve usage of rule based heuristics and procedures to generate new population of samples. This new population is generated by ”mutating” the vectors representing each sample. The combined population is then scored against itself using an appropriate fitness function and best performing set of samples from the populations are allowed to continue to the next iteration akin to natural selection. Such class of algorithms have proven to perform on par with leading machine learning approaches when the mutation heuristics and representation vectors are chosen appropriately.[61, 62, 63]

Deep generative models have been pivotal in driving novel methods for *de novo* molecular design methods. They are a class of methods that aim to capture the non-linear relation between the molecular structures and their properties. Different forms of data are transformed to and from each other using a series of linear transformation layers with non-linear activation functions between them. By capturing this information from a large dataset, the models try to emulate or learn the characteristic features of a molecule that lead to a certain kind of property or behaviour. Generative models have advanced considerably in the recent times with diverse and exciting applications in the fields of image processing,[64] natural language processing,[65, 66]and audio manipulation.[67]

Majority of deep generative models can be classified in three categories or a combination of those categories: Variational Auto Encoders (VAE), Reinforcement Learning (RL), and Generative Adversarial Networks (GAN). Figure 2.6 gives a high-level overview of the more recently used deep neural network architectures in the task of molecule generation. In cases where the motive is to optimize a given molecular property, there is a need of a gradient estimator which can help to improve the gener-

ator through back propagation. Neural networks require a gradient through which their parameters are updated, in anticipation that their performance too improves as the choice of loss reduces. This gradient estimator may act as a representative of simulations, experimental observation or classical property prediction algorithms. In a simpler approach, the property to be optimized could be modelled via another neural network and back propagated to the generator model.

Gómez-Bombarelli et al. [35] made an attempt using VAEs to generate novel molecules. The model was trained on SMILES representations of known chemical structure where it encodes the molecules into a lower dimensional vector space, and the decoder converts these continuous distribution of vectors back to discrete molecules. Jin et al. [40] proposed JT-VAE, in which the model generates a molecule in a two-step process. In this process, first a junction tree is constructed to represent the molecular substructure composition for the molecule. Then, a message passing neural network is used to decode the final molecular structure of the molecule. Graph-VAE by Simonovsky and Komodakis [36] is a graph based generative model which learns to generate the adjacency matrix of a molecule at once rather than step by step. Liu et al. [39] proposed a constrained graph variational autoencoder which uses a graph structured VAE to train a sequential generative model. Lim et al. [37] proposed a model based on the conditional variational autoencoder[68] for molecule generation. They demonstrated the utility of their method by controlling and imposing five target properties simultaneously on the latent space. They were also able to adjust a single property while keeping the others constant. Grammar Variational Autoencoder by Kusner et al. [38] represents SMILES strings as a parse tree from a context-free grammar. Using this parse trees representation for the VAE to encode and decode from directly ensures that the generated outputs from the VAE are always valid structures.

Another method for the generation of molecules is the use of GANs wherein the generator is competing against another discriminative model. The goal of the generator network is to model new data points close to the original distribution such that the discriminative model is not able to distinguish between the true and synthetic data better than random chance. Non-differentiability of the data and work around that limitation is the major point of interest in such methods. druGAN by Kadurin et al. [69] was one of the initial attempts at using GANs in the context of molecule generation. druGAN demonstrates a proof-of-concept by using generative adversarial autoencoders (AAE)[70] to identify molecular fingerprints which have certain anti-cancer properties.

In addition to the generation of molecules through these models, it is important to bias the process towards required properties. In case of VAEs, the presence of a continuous latent space representation for molecules opens up the avenue for the application of various global optimization algorithms like bayesian optimization and particle swarm optimization. These can be used to find the optimal molecule in the latent space which maximize/minimize the given properties[71, 72]. Blaschke et al. [73] combined the VAE and GAN approach for generation to create robust molecule generator and then used bayesian optimization to make sure that the generater creates molecules with specific properties.

In a work from Bagal et al., inspired by Generative Pre-Training (GPT) model that have been shown to be successful in generating meaningful text, the authors train a Transformer-Decoder on the next token

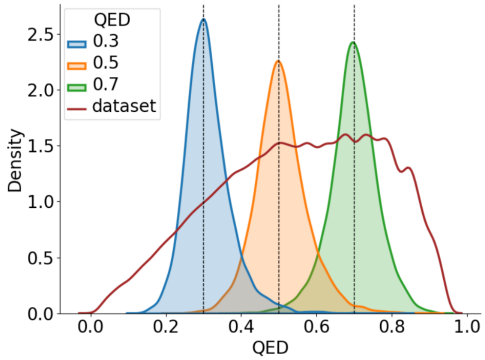


Figure 2.7: Distribution of the molecules generated by the generator based on the conditions imposed on QED vs the initial distribution of the dataset (Bagal et al. 23)

prediction task using masked self-attention for the generation of druglike molecules.[23] Additionally, they demonstrate that their model can be trained conditionally to control multiple properties of the generated molecules. An example of such a conditional generation is shown in Figure 2.7 where the generator is biased to generate molecules with QED close to a particular value.

ReLeaSE by Popova et al. [47] includes two deep neural networks: a generator (G) and a predictor (P). Initially, both the networks are trained independently with supervision from a separate dataset. In a later stage, the models are trained jointly using an RL method. The action space of the ”agent” i.e. the generative model is the set of possible SMILES notation alphabet and the state space is the set of possible strings in this alphabet. Rather than relying on any pretrained chemical descriptors, the models are trained on SMILES representation of molecules. The generative model consisted of a stack-augmented recurrent neural network, and QSAR models were used for the predictions. Goel et al. [22]

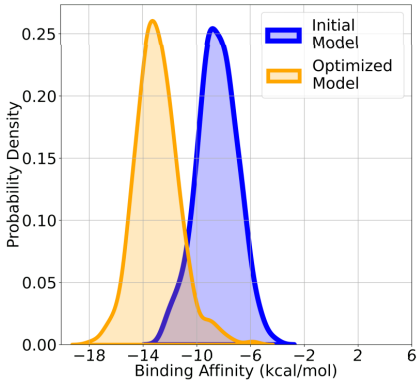


Figure 2.8: Distribution of binding affinity of molecules generated by an unbiased generator compared with the distribution of a generator biased for generating molecules with larger binding affinity to SARS-Cov-2 M_{pro}. (Goel et al. 22)

proposed MoleGuLAR, another stack augmented RNN based deep generative model which generates molecules with optimized binding affinity to a target. As an example, the change in distribution of the

molecules generated after optimizing for SARS-Cov-2 M_{pro} is shown in Figure 2.8. The pipeline is further extended for multi-objective optimization like logP, drug-likeliness, etc. There have been other studies with similar paradigm that use SMILES notation for molecular generation.[74, 75, 76]

ORGAN by Guimaraes et al. [43] extended the sequence based Generative Adversarial Network in SeqGAN[77] to include domain-specific objectives in addition to the discriminator reward in order to generate valid SMILES strings. By modelling the generator as a policy model in RL, this method bypasses the problem of discrete nature of molecular data since the model can be trained with gradient policy updates. The final reward of this is a combination of rewards returned by the GAN's discriminator and the reward generated by numerical function of the property prediction. This framework was tested using objective functions like solubility, synthesizability, and druglikeness. Another method, ORGANIC[78] explores the use case and performance of this model further by analysing how it performs with various other property criteria. Models like RANC, and ATNC uses differentiable neural computers which have explicit memory banks for generators.[79, 80]

The above models mainly used either learned representation vectors or SMILES strings of molecules as the descriptor for molecules. MolGAN[81] uses graph-structured data instead to generate molecules. Like others, the model uses RL objective that biases the model to generate molecules with specific desired chemical properties. Similarly, another method, Mol-CycleGAN[44] focuses on generating molecules or compounds that have a specific chemical scaffold while also optimizing for a property. LatentGAN by Prykhodko et al. [82] combines autoencoder with GAN for molecule generation. The GAN directly generates vectors in the latent space of the autoencoder and optimises the target properties. The model was tested in two scenarios: to generate general drug-like compounds and also target-biased compounds.

Another way to approach the problem is to train a pure RL agent to operate directly on a graph wherein the agent has to decide the addition of a new bond or atom in each action step amongst the predefined set of valid actions in the current state. You et al. [31] trained a general graph convolutional network based model for molecule generation to optimize domain-specific rewards. DeepGraphMolGen[83] extended GCPN by using Graph Convolution Networks to design a set of rewards to design small molecules. These molecules were generated to bind with dopamine transporters but not with norepinephrine. However, this model requires pretraining on specific datasets. Zhou et al. [46] introduced MolDQN a framework that combines chemistry domain knowledge and RL. Rather than using any kind of pretraining which could have reduced the search space, MolDQN learning from scratch based on its own experience. And unlike former methods, MolDQN also allows for multi-objective optimization.

Molecule Representation

A good representation is necessary for any ML method to perform well since the representation dictates what kind of information is available for the model to exploit and navigate the chemical space efficiently (Figure 2.9). Incorporating invariant and covariant properties of the system in the representation itself helps the models greatly since they don't have to waste training time on learning these concepts from scratch. A review by David et al. [84] discusses and analyses various representation of

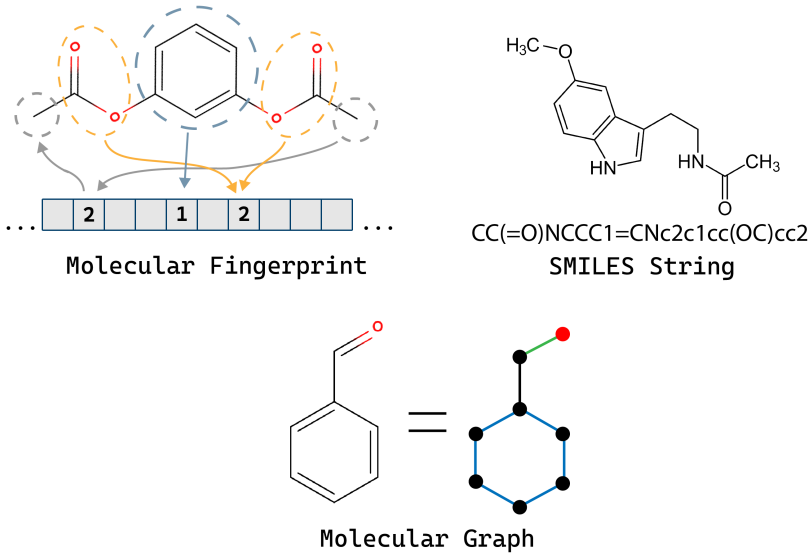


Figure 2.9: Overview of commonly used molecule representations

molecules in great detail. Concisely, majority of the representations used in generative models fall into one of the following categories: discrete string based,[22, 47, 74, 75, 76] continuous vector space,[82] weighted connected graphs.[31, 46, 85, 83]

2.2.2 Retrosynthesis

Retrosynthesis is the process of planning organic syntheses by finding possible readily available and simple precursors which on reacting produce the target molecule in one or more steps. This is done through breaking of bonds and functional group interconversion and retrosynthetic analysis has become an integral part of organic chemistry. Retrosynthesis formulates the organic synthesis process as an inverse problem by working backwards from the target molecule and systematically dissecting it to reach the simplest possible precursors as described by Corey.[86, 87] An example of such a retrosynthetic planning is shown in Figure 2.10.

Conventionally, this would require a chemist to use their knowledge of potentially thousands of reaction rules to find which possible precursors would lead to the given target followed by ranking them based on their feasibility. The process can also be done *in silico* with the reaction rules from the expert being translated into a program which can detect molecular substructures and the corresponding environmental information like functional group compatibility, stereoselectivity etc. Tools like Chematica[88, 89] (now Synthia) use hundreds of thousands of reaction rules curated by experts along with heuristics to terminate exploration of unpromising precursors to find reactants which are commercially available which can produce the desired product via single or multi-step reactions. However, manual accumulation of reaction rules is extremely labour intensive and dependent on the expertise

of the contributors. The rise of readily available, curated datasets has given a boost to the use of data driven methods for retrosynthesis.[90, 91]

2.2.2.1 Template Based

A wide variety of reaction rules (or templates) can be extracted from datasets like Reaxys[1], SciFinder[2] and reactions from chemical literature like the one created by Lowe from reactions in U.S. patent literature.[92] Template extraction is done in two steps: atom-atom mapping (AAM) followed by finding the reaction center. Plehiers et al. extracted the rules from InChI and SMILES representations of molecules by performing AAM using the Reaction Decoder Tool and then, finding the reactive center by identifying which atoms' environments changed during the reaction.[93, 94] Coley et al. defined strict SMARTS patterns to describe the reaction center, its neighbouring atoms and the corresponding functional group in the reactants and the product and finally merging the two into an overall retrosynthetic SMARTS pattern.[95] Law et al. took a different approach by first identifying the reaction center followed by extending it to encompass the relevant neighbouring atoms and these extended reaction centers are then clustered to get a generalized template.[96] These extracted templates are then applied to the given product to obtain the precursors.

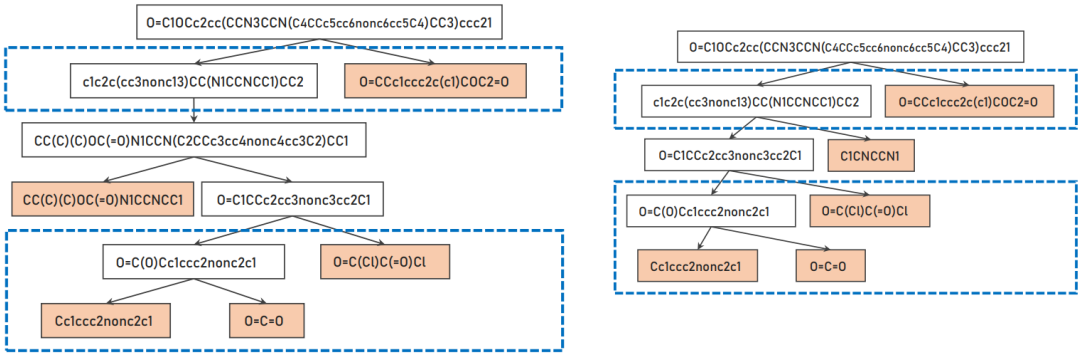


Figure 2.10: Example of retrosynthetic routes of a molecule as tree representation. The target molecule can be solved if it can be deconstructed to a set of readily available building blocks shown with a coloured background. Figure from Hong et al. 97 under Creative Commons licence

Using the extracted rules, there is a requirement for algorithms that can effectively search the retrosynthesis tree for the most promising paths and with the extensive amount of available data they can be driven by machine learning. For predicting the precursors of a single step reaction, studies by Ishida et al., Chen and Jung using graph convolutions have shown great promise.[98, 99] However, most products can rarely be derived from a single step and multi-step reactions should also be found for the task. The most popular algorithm that helps achieve this is Monte Carlo Tree Search (MCTS). Segler et al. used a variant of this algorithm in which they used three neural networks to first sample a template

[1] https://www.reaxys.com/
[2] https://scifinder.cas.org/

and apply it to the molecules such that the search goes in the most promising directions followed by predicting if the proposed reactions are feasible or not and finally estimate if the transformation is a "winning move" i.e. it leads to commercially available compounds to reward or penalize the neural networks.[100] In the work by Schreck et al. the authors proposed using a policy learned through reinforcement learning such that the policy minimizes the expected synthesis cost (a metric defined by the authors).[101] The open source AiZynthFinder software by Genheden et al. for retrosynthetic planning also uses a variant of MCTS.[102] The Retro* architecture by Chen et al. proposed a best first search algorithm using an "AND-OR" tree which can be used instead of MCTS.[103]

Template based approaches however, come with the caveat that any possible precursors will not be identified if the respective reaction does not belong to the extracted rules and it is not feasible to enumerate the exponential number of outcomes from the retrosynthesis tree. With the advances in machine learning especially its wide spread use in pattern recognition, template free models have also been developed which implicitly learn transformation rules between the reactants and the products.

2.2.2.2 Template Free

The SMILES string representation of molecules has its open grammar and semantics opening up the avenues for applying natural language processing based practices for a wide variety of tasks with retrosynthesis being one of them. Liu et al. modelled the retrosynthetic prediction task as a neural machine translation problem and used the seq2seq model to predict the reactant SMILES given the product molecules.[104, 105] With the advent of transformer models as the state of the art in translation tasks Karpov et al. proposed using it for single step retrosynthesis following which Zheng et al. and Kim et al. added different forms of SMILES correctors to make sure that the generated molecules are valid.[106, 107, 108] Mao et al. and Seo et al. combined information from molecular graphs with the transformer model to create an even more robust model.[109, 110] Lin et al. combined the transformer archtiecture with MCTS for the multi-step problem. They used a heuristic score at each reaction step to see how promising it would be to explore the subtree.[111] In the work by Schwaller et al., the authors trained a forward model to predict the products from reactants and calculate the reaction likelihood. Another transformer model was trained to predict the possible reactants that could lead to the product which there then ranked using the SCScore[113] and the reaction likelihood and the process was continued till commercially available precursors were found using the precursors as initial target molecules.[112]

Recently, newer approaches that take advantage of the best of both worlds i.e. templates as well as the ability of machine learning models to implicitly learn transformation rules called semi-template based algorithms.

2.2.2.3 Semi-Template Based

These algorithms use a two step process for finding precursors

- Using machine learning to identify the reaction center which gives information about the bonds that can break during the reaction. These bonds are then disconnected to get structures commonly referred to as synthons in literature.
- The reactants are then obtained through a series of transformations on these synthons.

One such study was reported by Shi et al. who treated the reactants and products as graphs. They used the R-GCN graph neural networks[115] to predict which bonds would break to produce synthons and then added new nodes and edges predicted using the same architecture to each synthon to complete their structures.[114] Similarly, Somnath et al. used the MPN[117] architecture for graph convolutions.[116]

Yan et al. proposed using Edge-enhanced Graph Attention Network for reaction center identification and the produced synthons were then converted into SMILES format which could be invalid in some cases. These invalid SMILES strings were then corrected using a transformer model.[118]

With a lot of open source platforms like AiZynthFinder, ASKCOS[3] and IBM RXN[4], the accessibility of AI/ML enabled retrosynthetic planners has improved significantly.

2.2.3 AI Powered Robotic Synthesis

Once a candidate molecule is generated and retrosynthetic logic has provided the recipe, the next step in the pursuit of complete automation is the automated synthesis of complex molecules. The two major objectives of automated synthesis are increasing reaction throughput commonly called high throughput experimentation (HTE) and increasing user autonomy so that the user input required becomes minimal. The latter objective would help in producing systems with the ability to synthesize molecules based on the provided retrosynthetic steps without necessarily leading to a high-throughput procedure.[119] A recent analysis of small drug like molecules found a lot of redundancy in the fragments present in them in terms of heterocyclic motifs.[120] In other words, the estimated number of drug like molecules is practically infinite, but the number of different fragments that form these molecules are less. Most autonomous systems are built *de novo* such that they provide high efficiency and flexibility but require great investment in terms of software and hardware.

With the obtained retosynthetic pathway for the target at hand, the conditions in which the reaction occurs are still missing. The use of machine learning has shown great promise for predicting the conditions as well. Gao et al. used fingerprints from the product and the reaction to predict the catalyst, solvents, reagents and temperature most suitable for the reaction.[121] In case of existing literature for reactions, Vaucher et al. used natural language processing to extract the experimental procedure from patents and scientific literature.[122] Aided with approaches like these, the pursuit for complete automation gets a major boost.

The system developed by Li et al. showed the possibility of generalized automated synthesis by using the same automated workflow for 14 distinct classes of molecules.[123] Steiner et al. developed the

[3] `https://askcos.mit.edu/`
[4] `https://rxn.res.ibm.com/`

”Chemputer” architecture, a generalized format for reporting chemical synthesis procedures that could link the procedure to physical operations. The authors also proposed a framework called ”Chempiler” to produce specific low-level instructions for the Chemputer architecture. It is responsible for finding paths between a source flask and target flask as well as address devices like hot-plate stirrers based on the vessels they are connected to. This architecture was then also applied to a physical platform and tested on three different drugs with extremely promising results.[124, 125] The AutoSyn automated synthesis system created by Collins et al. has been predicted to be able to synthesize 87% of FDA approved drugs with minimal manual intervention along with analytical monitoring during the synthesis process on a milligram to gram scale.[126] However, most automated synthesis systems require a set of instructions from the users which can then be followed but in order to close the loop these can be connected to a robust retrosynthetic planner.

Coley et al. split the automation process into two modules: synthesis planning and and robotic flow. The synthesis plan from the first module is converted to a chemical recipe file (CRF) which specifies the fluidic path to be constructed: locations of solutions, sequences of process modules, shutdown flow rates etc. However, the process is not completely automated and requires human intervention to load reagents following which the CRF is followed.[127] Another study in which machine learning was used to aid synthesis planning was presented by Granda et al. They used a machine learning model to predict the reactivity of a reaction mixture and the selected reaction was then automatically performed by a connected robot. The obtained results were then used as feedback to the machine learning model making it more robust as the number of reactions increased.[128] An important aspect of chemical synthesis is finding the appropriate conditions for a reaction to occur including temperature, solvents and more. Gao et al. [121] used machine learning to predict the catalyst, solvent, reagent and temperature for a given reaction. A study by Shields et al. [129] used bayesian optimization for finding the best conditions for maximum yield.

The RoboRXN platform by IBM[5] combines recent advances in cloud infrastructure, AI and chemistry to form an end-to-end autonomous system. In the industry, systems developed by companies like ChemSpeed and Syrris are making robust systems which can be employed in a wide array of reaction classes. The software for autonomous systems is also being developed with great rigour with platforms like ChemOS and ESCALATE becoming exceedingly popular.[130, 131]

We have moved very close to the goal of complete autonomy in synthesizing molecules exploiting well established synthesis methodologies but currently cost forms a major roadblock with systems costing thousands of dollars making them accessible to very few research groups in the world.[132]

2.2.4 Characterization of Molecules

Once the molecule that was designed *in silico* is realized *in vitro*, it is important to verify if the sample attained is actually the planned molecule. This is the problem of chemical characterization,

[5] https://research.ibm.com/science/ibm-roborxn/

wherein we can measure properties of a sample and have to determine its unknown molecular structure. This has been one of the persisting problems in chemistry and is often approached using spectroscopic techniques which measure how a molecule interacts with electromagnetic waves. Traditionally, experts manually identify the molecular structure from different kinds of spectra with highly domain specific knowledge which is time consuming especially in high throughput setting. The problem of chemical characterization as presented here is actually a kind of non-linear inverse problem. Forward model $y = f(x)$ here refers to the calculation of measured properties i.e. different kinds of spectra y given a molecule x. Whereas, the inverse problem is the task of elucidating the structure of an unknown molecule x from its experimentally observed spectra y.

Even today, majority of the computer based ways to characterize a sample using its spectra rely on matching the spectra of the unknown spectra with a database of already known spectra.[133, 134] The obvious drawback of such matching methods is that they restrict the usage to identifying only those molecules that are already stored in the database.

Infrared Spectroscopy (IR) is an analytical technique that reveals information about vibrational modes of movement of a molecule. Some vibrational modes in a molecule lead to change in the dipole moment and absorb light corresponding to those frequencies. IR spectra of a molecule is highly rich in information. The functional group region beyond $1500cm^{-1}$ can be used to identify the different functional groups present in a compound and the fingerprint region of the spectra $< 1500cm^{-1}$ forms intricate pattern that are used like a fingerprint to distinguish molecules.[135, 136] Wang et al. [137] use traditional ML algorithm support vector machine to do multi-class classification compounds from the OMNIC database based on their Fourier transform infrared spectra. The trained support vector machine identified 16 functional groups with prediction accuracy of 93.3%. Fine et al. [138] introduce a multi-label neural network to identify functional groups present in a sample using a combination of FTIR and MS Spectra. The work claims that their neural network reveals patterns typically used by chemists to identify standard functional groups. The model is also validated on compound mixtures while being trained only on single compounds.

Nuclear Magnetic Resonance (NMR Spectroscopy) is a spectroscopy technique that relies on the nuclei's magnetic properties to respond to externally applied magnetic field. The nuclei responds through signature electromagnetic waves that are then measured and recorded. There have been a few endeavours to solve the inverse problem of NMR Spectra to its original molecule in the recent times. Zhang et al. [139] used a tree-based search framework with a SMILES Generator to predict the structure from computationally generated ^1H NMR spectra. Their method included help from computationally expensive DFT Calculations to guide the tree and were able to predict structure from six out of nine given spectra. In a work by Jonas [85], a graph neural network is trained on molecular graphs with imitation learning. The NMR spectra is incorporated as per-node information in the molecular graph, and the molecule is built iteratively by adding edges based on the probabilities returned by the neural network.

The next chapter of this thesis proposes a Monte Carlo Tree Search based framework to approach this problem.[140] In this framework, value and prior models are pretrained using guided-MCTS runs

incorporating substructure information. This method was able to have the correct target molecule among its guesses for 93.8% of the molecules with < 10 heavy-atoms.

Mass Spectroscopy (MS) is another analytical technique that is used for chemical characterization. It measures the mass-to-charge ratio of ions present in a sample and presents it as a plot of intensity vs mass-to-charge ratio. Inverse problem of this kind can also be broken down into smaller parts wherein we try to find an intermediate representation g such that we learn the function $f \rightarrow g$. Hoping that the conversion from g to x is more convenient. Ji et al. [141] present a deep learning based approach, DeepEI. DeepEI elucidates the structure of an unknown compound from its Electron Ionization Mass Spectrum. DeepEI predicts molecular fingerprints from a spectrum and searches the molecular structure database with the predicted fingerprints. MESSAR by Liu et al. [142] takes a rule-based approach to identify and associate spectral features with substructures taken from databases with a goal of partial structure identification. Litsa et al. [143] proposed Spec2Mol, a deep learning architecture to be able to find the correct structure given a Mass Spectra of a molecule. Their approach is based on an encoder-decoder architecture where in the encoder learns the spectra embeddings, while a pretrained decoder tries to reconstruct SMILES sequences of the original molecule.

2.3 Summary and Outlook

The advent of modern machine learning algorithms has provided chemists with new tools in the pursuit of solving different inverse problems. The first task in this subset of inverse problems is to generate valid molecules, which was achieved by deep generative modelling methods such as RNNs, autoencoders, graph neural networks, and more recently, transformers. Once this is done, the next step is to tackle the actual problem of generating molecules that exhibit a specific set of properties. In order to achieve this, different algorithms like Bayesian optimization and reinforcement learning must be used to make the aforementioned generator models explore regions of the chemical space where molecules satisfy the given constraints.

However, generating a molecule *in silico* is not an end to itself since we still need a way to realize them. We need methods to find commercially available molecules that can be used to synthesize the molecule employing viable synthetic methodologies. Conventionally, for a new molecule, this would require domain knowledge to find possible reaction routes manually. This process has minimal throughput and depends heavily on the expertise of the scientists. The use of *in silico* methods to extract reaction templates essentially make retrosynthesis a pattern recognition problem for which machine learning has proven to be of great use in domains like natural language processing and computer vision. A collection of templates can be applied to new molecules to find their precursors, and different heuristics can be used to explore the most promising branches of the retrosynthesis tree.

A variety of alternate tools and methods to design molecules catering to their specific requirements are accessible. However, one could argue that the effort in automating the molecular design process has been disproportionately skewed towards just molecule generation and retrosynthesis. In contrast,

other vital tasks in the pipeline like automated robotic synthesis and chemical characterization are less explored. Research that uses spectroscopy data to solve the inverse problem of spectra to the molecule is sparse, and hence the problem could be considered an open one. The initial attempts at solving this problem using NMR and MS spectra show great potential, and the authors expect that this potential will be continued to be explored by many more studies in the coming years. Most of the work for IR spectra involves using the functional group region to classify molecules based on their functional groups. Even though Infrared Spectroscopy is known to be highly information-rich, with the fingerprint region of the spectra often being used to characterize samples in the lab, there are yet to be computational methods that aim to learn and exploit those relations to determine the target structure. Thus, such an application to relate IR spectra directly to molecular structures would be an exciting avenue for further research. Since each of the spectra discussed in this *highlight* reveals a different kind of information about a molecule, a method combining different kinds of spectra to evaluate the structure of a sample would also be of great promise in the molecular design pipeline.

Unlike other subtasks in this *highlight* which mainly depend on computational resources and novel architecture for progress, the high cost of robotic equipment and the need for hardware expertise makes research in AI-assisted robotic synthesis inaccessible to a large section of the community. With speculations that complex robots would only become cheaper and more accessible, it may not be a distant dream that this would allow more and more research groups to conduct leading research in this area of AI-assisted robotic synthesis.[144, 145]

The speed and throughput with which the problems mentioned in this *highlight* are being solved today did not seem possible at the beginning of the decade. However, the availability of new algorithms and reduction in costs of hardware like GPUs that work in conjunction with each other has helped open up many possibilities in this domain. Democratization of information and ease of accessibility of leading research to the general population has greatly helped the scientific community develop and share their work on these problems. Such rapid progress and development is expected to continue as time progresses and would extensively drive the discovery of novel molecules and their application.

Chapter 3

Deep Reinforcement Learning for Molecular Inverse Problem of Nuclear Magnetic Resonance Spectra to Molecular Structure

3.1 Introduction

Spectroscopy in general has played a significant role in diverse applications such as drug discovery, protein structure determination and material discovery. Nuclear Magnetic Resonance (NMR) spectroscopy is one of the most crucial and versatile methods for chemical characterization. It is an analytical technique based on the nuclei's magnetic properties that either have an odd mass number or an even mass number with an odd atomic number. Nuclei with non-zero spin $\overrightarrow{S}$ would always have a non-zero magnetic dipole moment, $\overrightarrow{\mu}$. NMR relies on this for the nuclei to respond to electromagnetic waves as perturbations in the presence of an external magnetic field. In addition to small organic molecules, NMR spectroscopy is a critical method to obtain high-resolution information about proteins, DNA, and RNA[146, 147]. It can also be used to obtain knowledge of energy minima and barriers by observing conformational dynamics of proteins [148]. This can be pivotal in the process of drug discovery.

The ^{13}C NMR spectra measures the properties of individual nuclei and consists of peaks that correspond to each carbon atom present in the molecule. The peak position (chemical shifts) and the peak splits (spin-spin coupling) are dependent on the local environment of that atom. Usually in labs, experts manually identify the molecular structure from the NMR spectra using highly specific domain knowledge. Till today, most computer-based methods to verify the structure of a sample from its NMR spectra rely on matching the spectral data with a database of already known spectra [133, 134]. These methods restrict the usage to identifying only those molecules that are stored in the database.

The problem concerned here is a non-linear inverse problem. Forward model $y = f(x)$, in this context, refers to the task of calculating the NMR spectra y, given a molecule x. Whereas, the inverse problem refers to drawing conclusions about an unobserved molecule x from its experimentally observed NMR spectra y. One of the first attempts in literature at recognizing and modelling this problem as an inverse problem was done by Jonas 85. (Figure 3.1).

The forward problem f for NMR spectra is relatively well studied with many methods ranging from

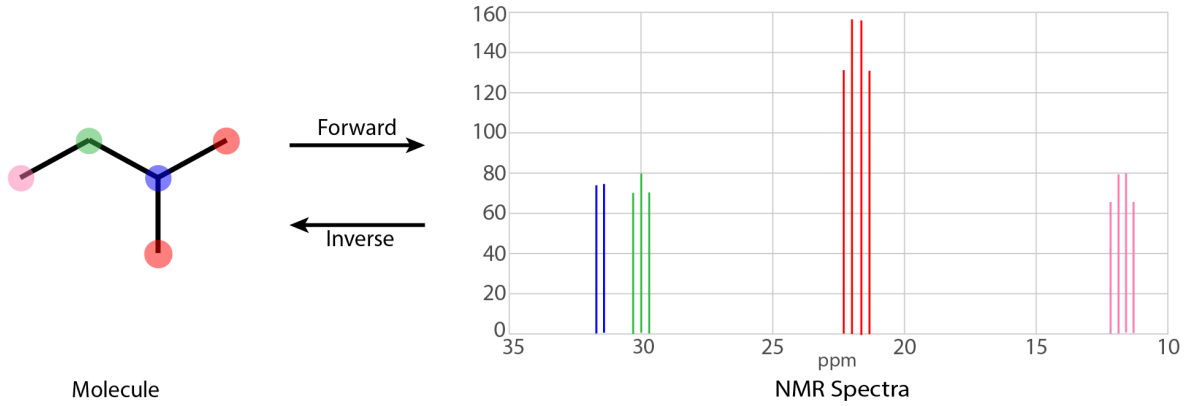


Figure 3.1: Schematic representation of the Forward and Inverse problem of NMR Spectra.As an example, 2-methyl butane and its NMR spectra is given. Each carbon and its corresponding peak is indicated by the same colour.

quantum mechanical calculations and density functional theory [149] to deep learning [150, 151] to solve the task. Other empirical methods such as featurizing the neighbourhood of a nuclei and then matching it against a database of known motifs to predict its shift are also common[152, 153].

Recently, there have been many significant studies on the use of modern deep learning and RL methods to solve problems in chemical sciences ranging from prediction of properties of molecules to *de novo* molecule generation with optimized properties[46, 31]. There are various high-throughput combinatorial methods to generate organic molecules with desired properties that are well known and essential in the process of drug discovery[52, 53, 22, 23, 47]. In such a workflow, it would be of great help to have a framework to verify the structures of samples generated *in situ* based on easily acquirable spectral data in a high-throughput manner.

In this work, an effort has been made to determine the structure of a molecule, given its NMR spectra and molecular formula. There have been a few endeavours to solve this inverse problem. Zhang et al. [139] used a tree-based search framework with a SMILES generator to predict the structure from computationally generated ^{1}H NMR spectra. Their method included help from computationally expensive DFT calculations to guide the tree and were able to predict structure for six out of nine given spectra. In a work by Jonas [85], a graph neural network is trained on molecular graphs with imitation learning. The NMR spectra is incorporated as per-node information in the molecular graph, and the molecule is built iteratively by adding edges based on the probabilities returned by the neural network. Their work was tested on molecules with up to 32 heavy atoms.

In this work, we use a combination of online Monte Carlo Tree Search (MCTS) [154] and a set of offline trained Graph Convolutional Networks [155] to navigate through the chemical space and find the correct molecular structure of a given target ^{13}C NMR spectra.

3.2 Methods

To reiterate, the problem is defined in the following way: given ^{13}C NMR spectra of a molecule consisting of each carbon's shift and split values and the molecular formula, identify the structure of the molecule. The process of solving the problem is modelled as a Markov Decision Process (MDP) wherein the molecule is built iteratively from scratch by adding atoms and bonds to the current structure at each step. In this section, different components and details about the proposed framework are explained. The first part contains information about the dataset used. Then the text gives details of the reinforcement learning algorithm used followed by information about the neural networks that aid the RL algorithm. Finally, the last paragraph of this section introduces a novel methodology of training prior and value networks.

3.2.1 Dataset

We use nmrshiftdb2,[156] which is a database for organic structures and their experimentally observed ^{13}C NMR spectra. In this work, we only consider organic molecules that have less than 10 non-hydrogen atoms (C, O, N, F). Charged molecules and radicals are also excluded. Thus, the dataset comprises a total of 2134 molecules with experimentally obtained chemical shift and split values of ^{13}C NMR spectra.

3.2.2 Reinforcement Learning (RL)

In this subsection, we attempt to define the state space and action space for the MDP comprehensively. This is followed by information about the agent which chooses the appropriate actions at a particular state. Choosing an accurate and appropriate measure of reward is essential for any RL algorithm to perform well. We use a forward NMR prediction model to formulate a reward function defined at the end of this subsection.

3.2.2.1 State Representation

- The current state in the search process is represented as a **Molecular Graph**.

- Each atom in the target molecule is present in the current state as a node. The graph of the current state has $n - s + 1$ components, where n is the total number of atoms in the target molecule, and s is the number of atoms present in molecule of the current state. Out of these $n - s + 1$ components, one is a connected component representing the molecule of the current state, and the rest of the $n - s$ components are individual atoms that may join the current molecule by addition of new bonds later on. Here a component is a sub-graph which doesn't have any outgoing edges to the rest of the graph.

- **Featurization of the target NMR:** Each NMR peak is assigned to a carbon in the beginning when the state consists only of individual nodes and no edges. The node feature of an atom consists of the one-hot encoding of the atomic number of the element that the node represents and the current valency of that atom that is available for further addition of bonds. A Gaussian, with the peak of the assigned shift centred at the chemical shift value and $\sigma = 2$, is discretized into 64 bins. This feature is then appended to the node feature.

- Since this work uses ^{13}C NMR spectra as an input, it is certain that the target molecule contains at least one carbon atom. Hence, without the loss of generality, we choose to start building our molecule from a molecular state containing a single carbon atom i.e. S_0 is just a carbon atom.

3.2.2.2 Action Representation

In this work, we formulate a fixed-dimension action space in which each action signifies the addition of an edge between any two nodes in the graph. The environment ensures the validity of these actions by checking for the following conditions:

- At least one of the endpoints of the newly added edge must belong to the sub-graph containing molecule of the current state.

- The addition of this new edge must obey the chemical rules of valency for each atom. If the valency due to connection with other heavy atoms is not enough to complete its octet, it is implicitly assumed that the rest of the valency is satisfied by hydrogens. These hydrogens are not taken as nodes in the molecular graph.

- The edge must not lead to formation of a ring with four or three atoms.

- The edge must not lead to a bond within an already present ring. [1]

3.2.2.3 Agent

Since the problem is formulated as a Markov Decision Process, we are left to decide on a planning algorithm that would use some prior knowledge about the problem and explore various branches of the search tree before taking action a on a state s. A typical RL algorithm has two components: an Agent, and an Environment. The task assigned to the "agent" is to choose an action given the current state. On the other hand, the role of the "environment" is to simulate the action which was chosen by the agent and return the reward for the action which was taken.[157, 158] One such algorithm is Monte Carlo Tree Search (MCTS) (Figure 3.2). MCTS performs one of the four following steps repeatedly:

[1] Note that this restriction does not prevent the formation of bicyclo and spiro compounds. It just guides the formation so that the smaller ring is formed before the larger ring. Doing so proved to be helpful in the initial experiments since this helps prune some redundant branches of the tree search.

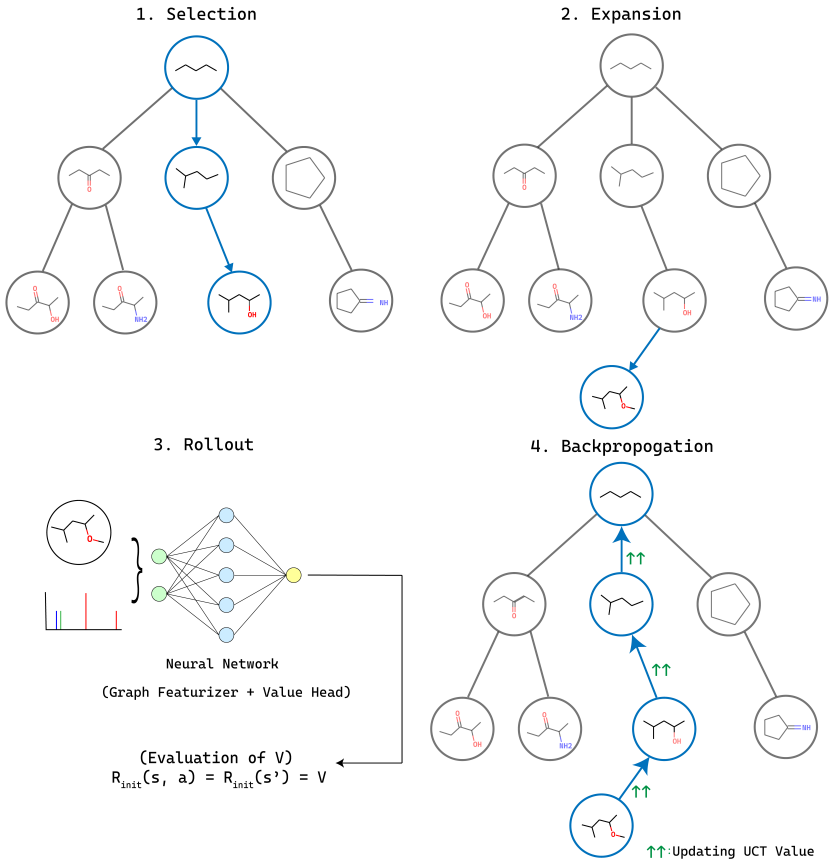


Figure 3.2: Monte Carlo Tree Search: A heuristic search algorithm where each node in tree is a state of the environment. One of these 4 steps is taken at each step to navigate the search space. **1. Selection:** New bonds are added to the root based on UCT values until a leaf node is reached, **2. Expansion**: A new node is added to the leaf node after environment checks the validity of new node.**3. Rollout:** Value neural network evaluates the value of the newly added node. **4. Back-propagation:** The tree then back-propagates the new information to update the UCT values of all the nodes till the root node.

1. **Select :** In this stage of MCTS, the tree is traversed from the root according to the UCT (Upper Confidence Bound for Trees) values at each level until it reaches a leaf node. The UCT value at any state is calculated based on the following formula:

$$UCT(s,a) = Q(s,a) + c * \pi_{\text{model}}(a|s) * \frac{\sqrt{N(s)+1}}{n(s,a)+1}$$

Where s: Current state, $Q(s,a)$: Mean Action Value estimate $Q(s,a) = \frac{W(s,a)}{n(s,a)}$, $W(s,a)$: the cumulative of all returns $R(s',a')$ till the leaf node, $\pi_{model}(a|s)$: Prior probability by the policy network, $N(s)$: Number of times state s has been reached, $n(s,a)$ is the number of times action a was taken from state s , and c is the constant with which one can manipulate exploration vs exploitation ratio. The form of UCT value used in this work is inspired by Moerland et al. [159], which proved to improve the performance of cases with asymmetric trees.

2. **Expand:** Once a leaf node s'_L is reached by the tree search, the tree is expanded by addition of a new leaf node s_L. The environment simulates this action and ensures its validity and also returns an intermediate reward.

3. **Roll-out:** In a typical MCTS, the initial value of the new leaf node is estimated using a series of random rollouts from the leaf node s_L. Due to computational limitations, this work uses a value neural network $V_{\text{model}}(s)$ to estimate the value function.

4. **Back propagation:** After estimating the value of the newly added leaf node, $R(s, a)$ of the whole backward trace is updated through back propagation which in turn updates the UCT value of intermediate nodes belonging to this trace.

$$R(s_i, a_i) = r(s_i, a_i) + \gamma R(s_{i+1}, a_{i+1})$$

The above four steps are repeated for $nmcts$ number of times. Then, a real action a_t is taken by the environment based on the policy of the tree. The tree's policy probability is determined by the visitation count of all the actions at the root node s_0.

3.2.2.4 Role of the Forward Predictor

While the inverse problem is defined as the task to determine the molecular structure from the spectra, it naturally follows that the forward problem is that of calculating the NMR spectra given the molecule and its structure. Here, a forward NMR prediction model [150] is used for the following:

1. **For intermediate reward:** Typical MCTS applications also have an intermediate reward returned by the environment for each action. The step reward is calculated based on how close is the current state to the target molecule. The forward model predicts the NMR spectra of the current state and the reward is defined by:

$$r(s, a) = r(s') = 2 * (\frac{1}{2} - WS(s'))$$
$$WS(s') = \text{First Wasserstein Distance} = l_1(S_T, S_C)$$

$$l_1(u, v) = \int_{-\infty}^{\infty} |U - V|$$

where U and V are the CDFs for the distribution of some random variables u and v [160], S_T is the NMR spectra of the target molecule, and S_C is the NMR spectra of the current moleculess'. The reward $r(s') = WS(s')$ is returned whenever the current state s' is known to be a terminal state. Otherwise, $r(s') = 0$ is returned.

2. **For the Scoring Function:** Each episode performed by the agent returns one prediction of what the target molecule is. Since MCTS has some element of randomness, all guesses made by the agent are not the same. In such cases, after running the agent for a predetermined fixed number

of times, all the unique guesses are ranked against each other by the means of the reward function discussed above. Then, the guess which returns the highest reward is taken as the final prediction.

3.2.3 Neural Architecture of the Prior Policy and Value network

There are three modules of neural network used in this work.

3.2.3.1 Graph Featurizer:

Graph Featurizer: This module uses Message Passing Neural Network[155, 161], which provides a formulation for supervised learning on graph structured data. Consider a **molecular graph** $G(V, E)$ with node features x_v (having information about the current state and also the target NMR spectra) and edge features e_{vw}. The features of each node at time step t are represented as h_v^t, initialized to x_v at $t = 0$. The features of nodes are updated for 3 time steps using messages m_v^{t+1} in the following way:

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

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

$$F_v = g(x_v, h_v^t) = x_v + h_v^t, \forall v \in V$$

where $N(v)$ is the set of neighbouring nodes of v. M_t and U_t are the message function and vertex update function respectively. The function g is simply taken to be vector addition in this work. F_v is the final atomic feature for the node which has information about the atomic properties, the local environment, and also the target NMR shift value that was assigned to this node. The feature (F_v) generated here will be further used by the policy neural network $\pi_{\text{model}}(a|s)$ and value network $V_{\text{model}}(s)$.

3.2.3.2 Policy Head:

Policy Head: N nodes form $\binom{N}{2}$ pairs, each representing a possible edge. For each of these pairs, let the feature vector of the pair be the concatenation of the feature vector of the two nodes concerned. This pair's feature vector is then passed through two fully connected layers to obtain a $3-$tuple representing the possibility of single, double, and triple bond between this pair.

3.2.3.3 Value Head:

Value Head: All the node features received from the graph featurizer are then sum-pooled to attain a molecule-level feature vector which has information about both the current molecule and the target NMR. This molecule level feature is then passed through two fully connected layers to finally predict the value $V_{model}(s)$ of the current state.

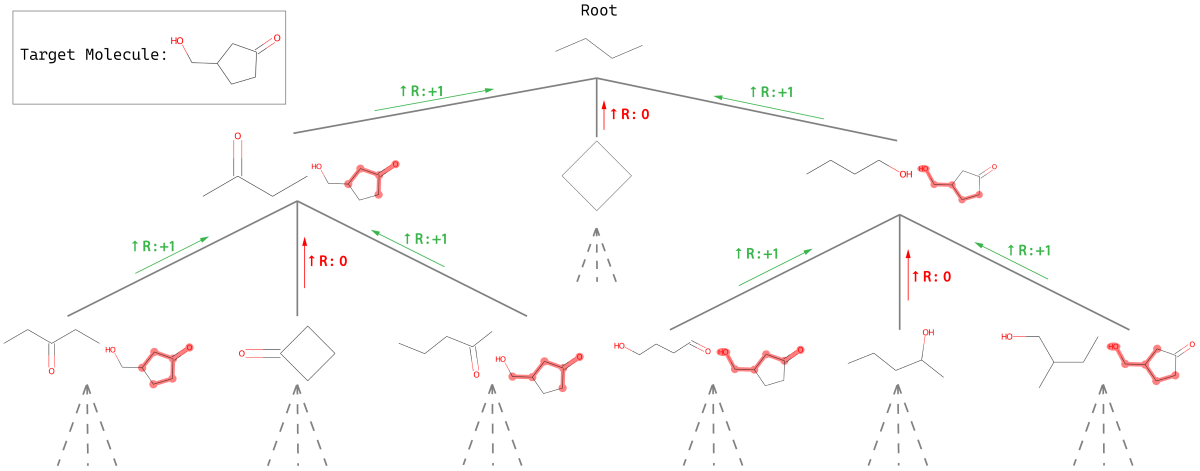


Figure 3.3: Training Methodology : An example search tree while in the training mode on how subgraph isomorphism is used to make dataset for the neural networks to train on. The target molecule is shown on the top left . Assuming that the current state in the environment is n-butane, we see a possible state of the search tree. Each node in the tree is also accompanied by an illustration showing how it is a subgraph of the target molecule. Since this can be evaluated when the training mode is on, this is used to return intermediate reward $r(s, a) = 1$ when the current state is subgraph isomorphic to the target state, otherwise $r(s, a) = 0$.

3.2.4 Training and Testing Methodology

3.2.4.1 Guided runs for training of the neural networks

While in the training mode, the environment has access to not only the NMR spectra but also the structure of the target molecule. This can be used to guide the tree by giving a strong positive reinforcement in the form of $r(s, a)$ (Figure 3.3). The tree policy (derived from visitation counts of the actions) and approximation of $Q(s, a)$ hence obtained is used as the training dataset for the prior policy neural network $\pi_{\text{model}}(a|s)$ and value network $V_{\text{model}}(s)$. When any action a at state s leads it to state s', i.e $s \xrightarrow{a} s'$, then:

$$r(s, a) := 1 \quad \text{iff} \quad S(s_t, s'), \quad \text{else} \quad 0$$

where $S(s_t, s')$ is Boolean function that returns True iff s' is subgraph isomorphic to the s_t. This work employs rdkit[162] to check whether the state s' is a sub-graph of the target molecule s_t. With training mode on, the model was run on system with Intel Xeon E5-2640 v4 processor and Nvidia GeForce GTX 1080 Ti GPU for 23 hours to collect experience and train the neural networks. Five models were trained on different crossvalidation training sets.

3.2.4.2 Using the Split Information to prune the trees during MCTS runs

Each shift value in the dataset is accompanied by a split value as well. The split value is a categorical variable that belongs to one of $\{S, D, T, Q\}$, and it is dependent on the number of hydrogen atoms that

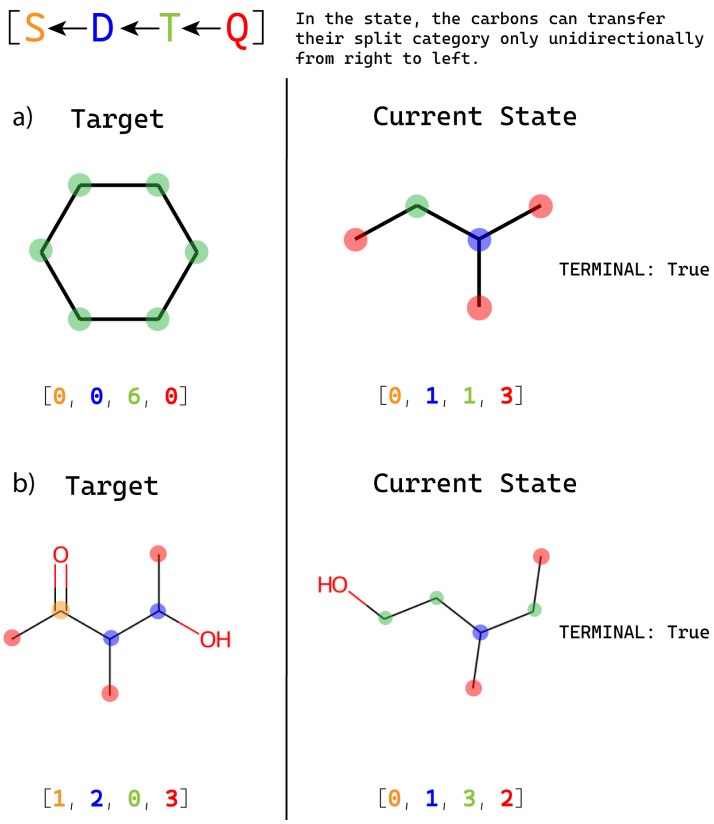


Figure 3.4: a) Target state of cyclohexane and current state of 2-methyl-butane along with their splitvectors b) Target state of 4-hydroxy-3-methylpentan-2-one and current state of 3-methylpentan-1-ol along with their splitvectors

are attached to the carbon. A quaternary carbon (no hydrogen attached) leads to a singlet (S) split, a tertiary carbon (one hydrogen attached) leads to a doublet (D) split, a secondary carbon (two hydrogens attached) leads to a triplet (T) split, and a primary carbon (three hydrogens) leads to a quartet (Q) split. Let splitvector be the vector that stores the information about the number of carbons of each split kind in the current state. Since the only action possible in the modelled MDP is that of addition of an edge(decreases the number of implicit hydrogens), note the following two invariant properties:

- The sum of values in the splitvector would remain constant for states with only 1 connected component since the total number of carbons can't increase.
- With addition of bonds, the kind of split made by a particular carbon can only move in the direction:

$$S \leftarrow D \leftarrow T \leftarrow Q$$

As a consequence of this, certain states can be flagged as terminal states if it is known that they can never lead to the target molecule based on the following criteria:

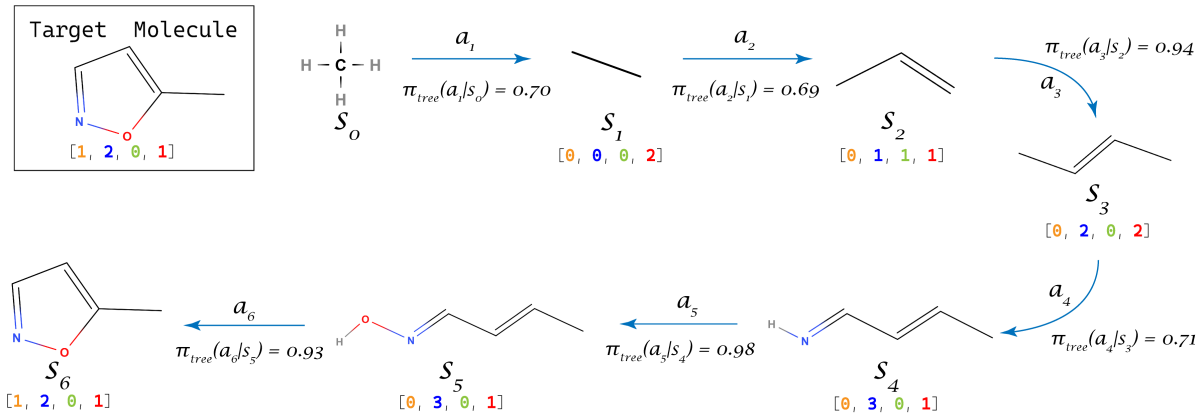


Figure 3.5: An example run for the target molecule CC1=CC=NO1 with $nmcts = 1000$, $\pi_{\text{tree}}(a_i|s_j)$ represents the probability of taking action a_i according to the policy returned by the MCTS Search with state s_j as the root. In the above figure, each state s_j is also accompanied by the splitvector of that state.

- When the number of quaternary carbons in the current state becomes lower than the number of quartet splits in the target spectra.
- When the number of singlet carbons in the current state becomes more than the number of singlet splits in the target spectra and so on.

For example, in Figure 3.4 a, the agent can safely terminate search through this branch since once a duplet has formed in the current state, that carbon can never be transformed back to triplet or quartet and we know that the target molecule does not have any duplet or singlet carbon. Similarly, in Figure 3.4 b, the agent can safely terminate since the number of quartets in the current state has gone below the number of quartets in the target molecule and there is no way to produce new quartet carbon atoms. These chemistry guided conditions greatly prune the search tree and prevents the tree from exploring branches that can lead to the incorrect structure.

3.3 Results and Discussion

3.3.1 Accuracy of the Forward model

The forward model used in this work was trained on nmrshiftdb2 dataset [156] as included in the original work by Jonas and Kuhn [150]. The mean absolute error obtained for the prediction of the shiftvalue per peak for the predictor was 1.374 ppm.

3.3.2 Crossvalidation

Figure 3.6: Accuracy over 5-fold cross-validation with $nmcts = 1000$

The total dataset of 2134 molecules was randomly split into 5 equal groups. In each of the five experiments, one of the groups was chosen as the hold-out test dataset and the model was trained on the remaining four groups. For each molecule, the agent made a number of guesses depending on how many episodes it ran. There were 20 processes initiated, with each of them running an episode. An example of one of such episode runs in illustrated in Figure 3.5. As shown in Figure 3.6, on an average, the agents guessed the correct structure of the molecule of the target spectra 93.8% of the time. All the guesses of the agent are then ranked based on the scoring function discussed in the earlier section. The Top1 ranked structure among the guesses was the target structure 57.2% of the time. Accuracy for Top3, Top5, and Top7 of the scored guesses can be seen in Figure 3.6. Figure 3.12 shows the Top3 guesses made by the framework ranked by the criterion mentioned earlier for a couple of examples where the Top1 ranked molecule is the correct target structure and Figure 3.13 shows examples where the Top1 ranked molecule is not the correct target structure.

3.3.3 Effect of nmcts on the Accuracy

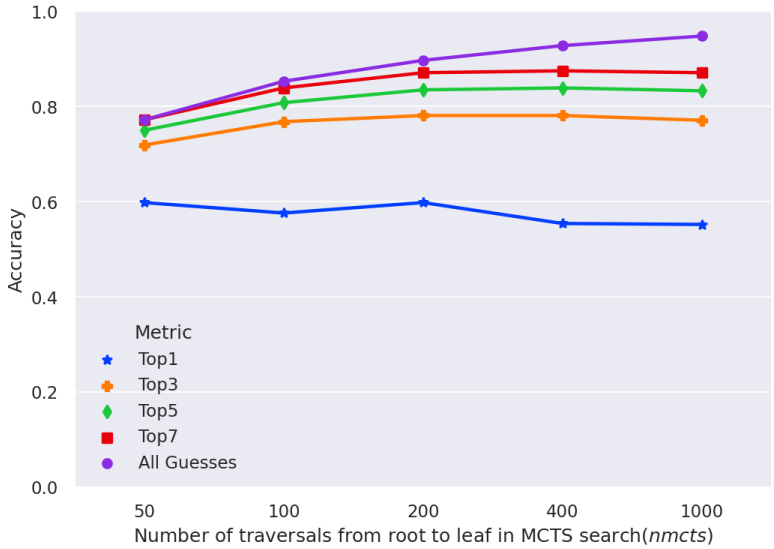


Figure 3.7: Effect of $nmcts$(Number of traversal from the root to leaf in MCTS search) on the various metrics of accuracy

$nmcts$ is the number of times that the search tree is traversed from the root to leaf node to explore different branches before making a true action in the current state. As expected, it can be observed in Figure 3.7 that the net accuracy improves as $nmcts$ increases.
It is also seen that the trend for Top1 accuracy is not the same as others and it actually decreases with increase in $nmcts$. This can be reasoned with the fact the increasing $nmcts$ increases the exploration of the chemical space and more potential candidates are scored against the current structure. This downward trend reveals that a better scoring function would improve the TopN accuracy of the agent since it would be able to rank the candidate guesses in a more accurate way.

3.3.4 Relation with the Forward Model

Even when the agent has the correct structure among its guesses, sometimes the scoring function ranks it lower than other guesses made which reduces accuracy. Since the scoring function is dependent on the pretrained forward model, a more capable forward model is expected to increase the accuracy of the framework. In the event that a better, albeit computationally expensive scoring function is deviced, the overall practical accuracy can still be improved while being even more time efficient. This can be done by scoring only the TopN guesses of the agent with the time intensive scoring function. As can be seen in Figure 3.7, target spectra's correct molecular structure is present in the Top7 of the guesses $> 85\%$ of the time for all the runs with $nmcts > 200$. In such a scenario, we can determine the correct

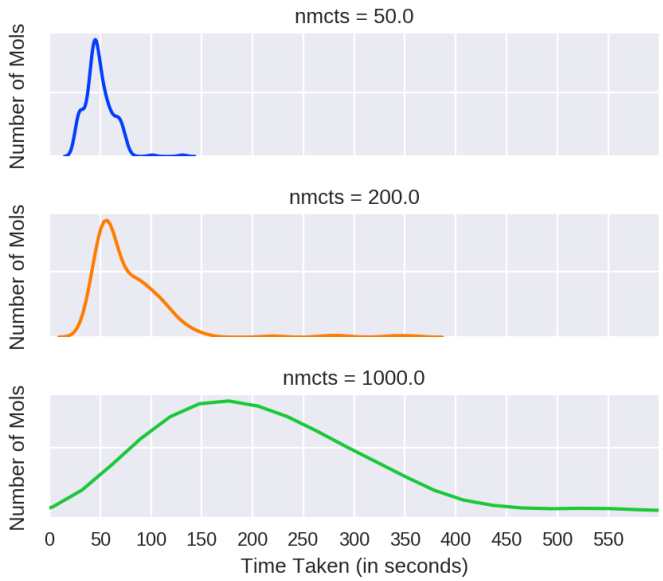


Figure 3.8: Effect of $nmcts$ on the time taken by the system to predict a molecule

structure for an NMR spectra by scoring only 7 structures while the MCTS search takes less than 100 seconds for most of the molecules.

3.3.5 Holdout Experiment with Training Data

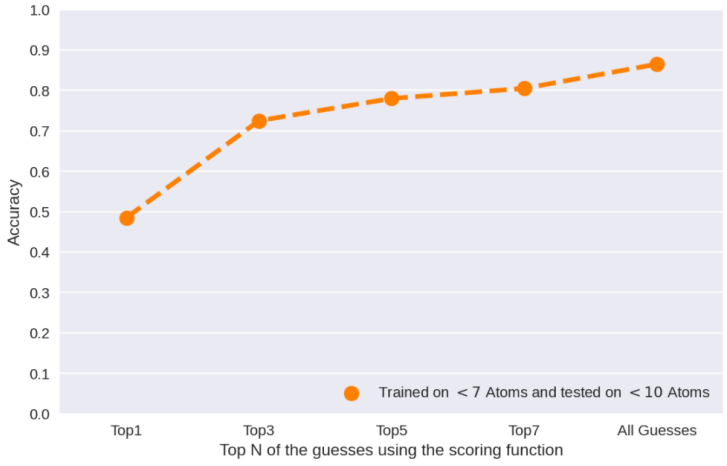


Figure 3.9: Accuracy when the model is trained of molecules with < 7 atoms and tested on molecules with < 10 atoms.

In another experiment, molecules with < 7 non-hydrogen atoms were filtered from the dataset. After running the agent on these filtered set of molecules with training mode on, the agent was tested on 200 randomly sampled molecules with ≥ 7 and < 10 heavy atoms. The result of this experiment is plotted

in the Figure 3.9. However, the system performs well on the class of data that it was never exposed to before, by guessing the correct structure 86.5% of the time.

3.3.6 Time difference between correctly guessed and incorrectly guessed molecules

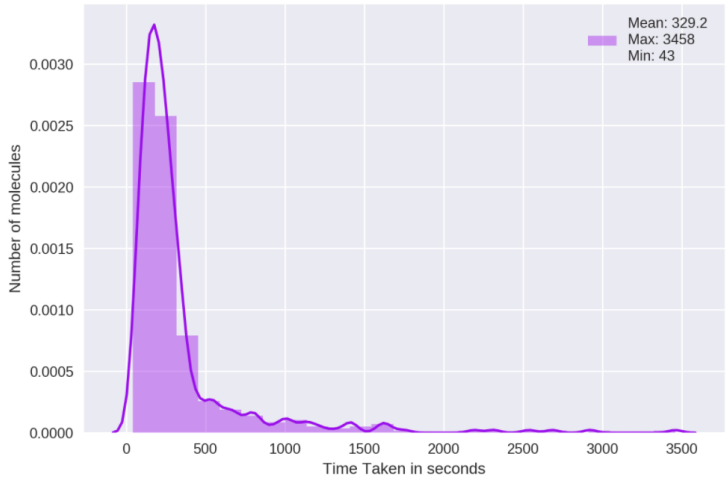


Figure 3.10: Histogram of time taken for the model to run on each molecule for $nmcts = 1000$)

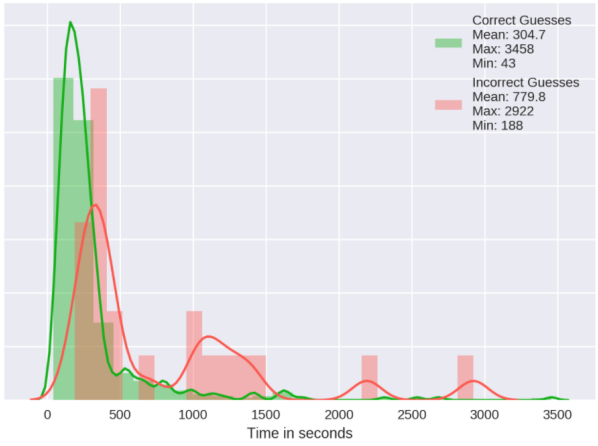
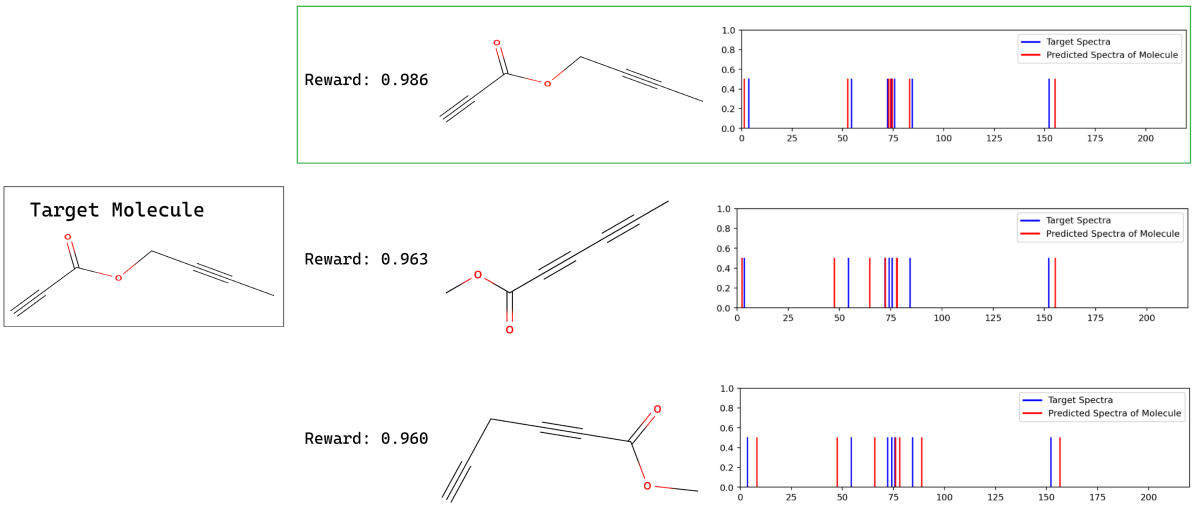


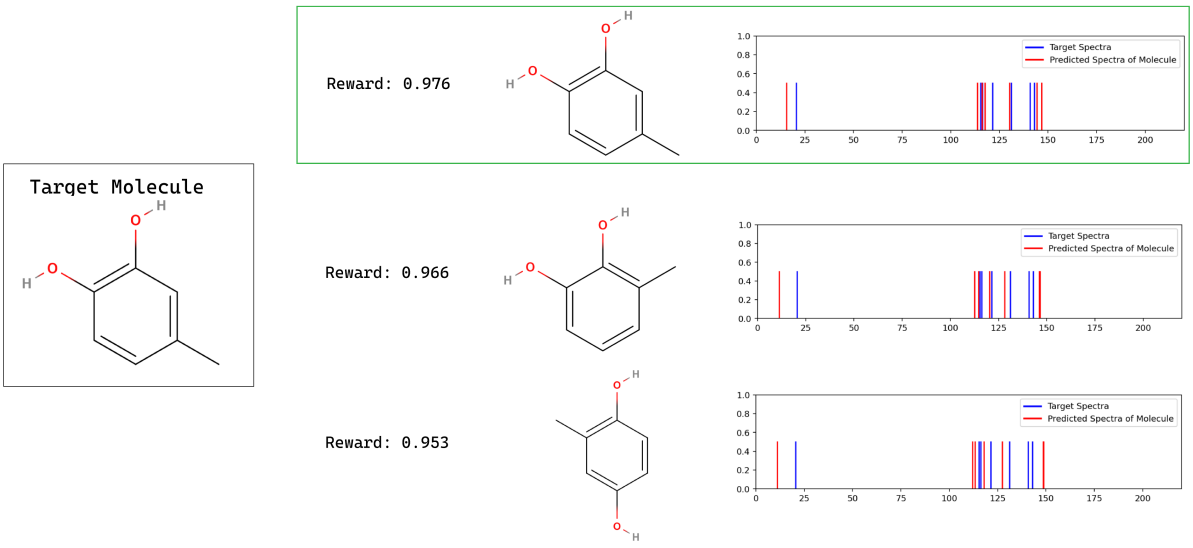
Figure 3.11: Time taken for a molecule when it is guessed correctly and incorrectly ($nmcts = 1000$)

The histogram of the time taken for the agent to run all the episodes for a molecule can be seen in Figure 3.10. On an average, it takes ≈ 330 seconds for the agent to make all its guesses for a target NMR spectra. All episodes are run within 300 seconds for 71.8% of the molecules and within 600 seconds for 88.5% of the molecules. It is observed in Figure 3.11. that the mean time taken for all the episodes for a molecule that is guessed correctly is ≈ 305 seconds, whereas the mean time for molecules that are guessed incorrectly is ≈ 780 seconds. This difference of distribution can be used to have more reliable predictions and improve the potential practical use-case of this framework. Stopping the search at a

threshold time can improve accuracy for predicted molecules while also saving computational expense. When the framework makes predictions for all the molecules, i.e without any threshold time, the correct structure is among the guesses made for 94.8% of the molecules. Having a threshold time of 300 seconds leads to the framework making predictions for 72% of the molecules and timing-out for the rest of the molecules. The correct structure is among these guesses for 99% of the molecules. Similarly, when the threshold is set to 1000 seconds, the framework makes predictions for 94% of the molecules. The correct structure is among the guesses for 97% of the time.

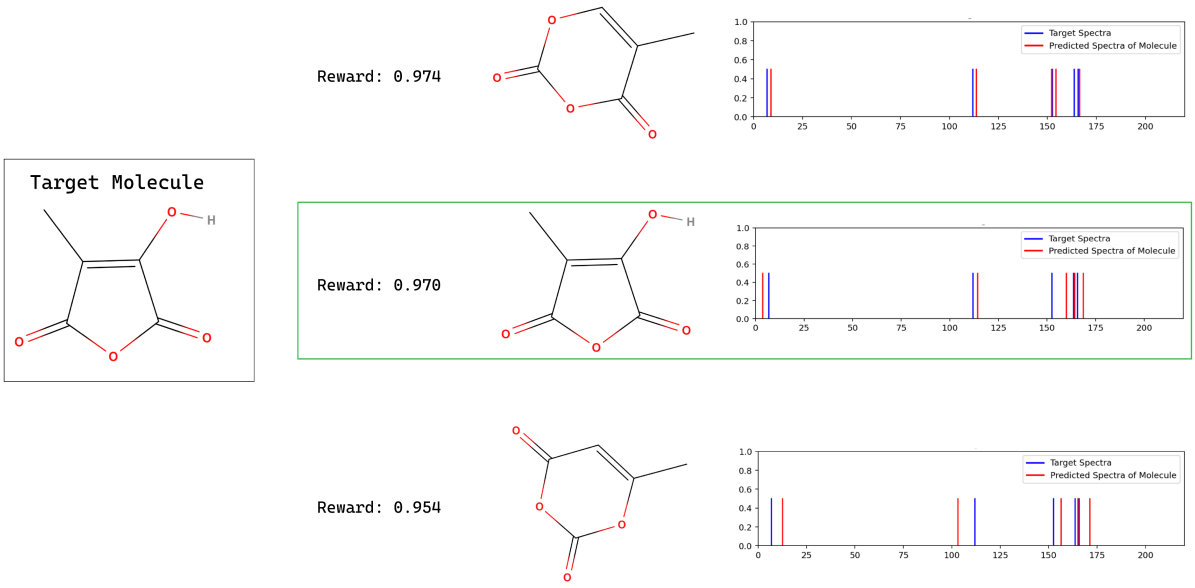


(a) Example 1: Top3 ranked guesses by the framework for experimentally observed NMR spectra of C#CC(=O)OCC#CC

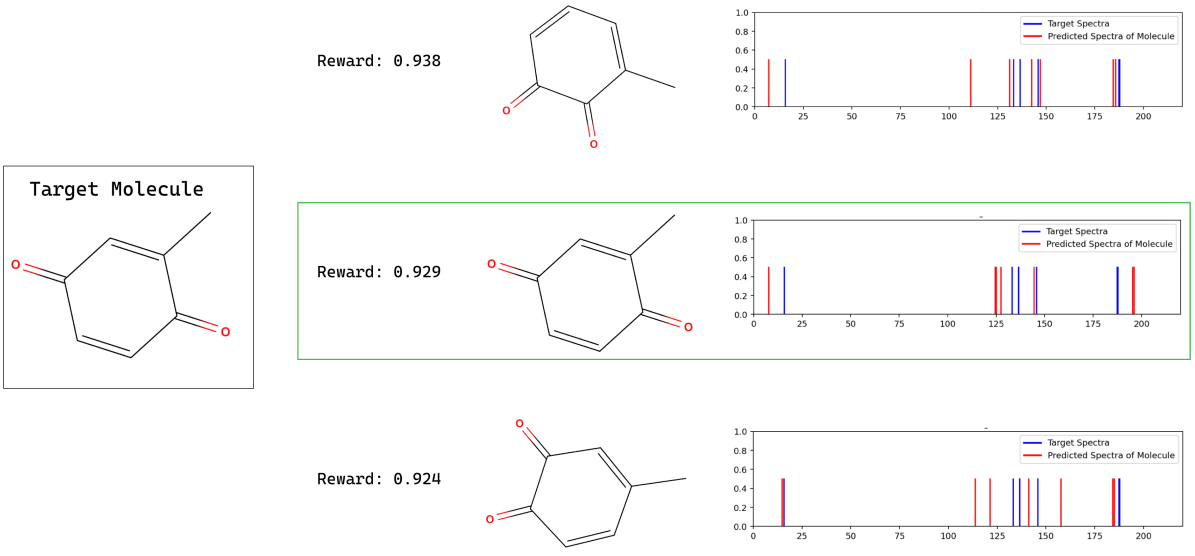


(b) Example 2: Top3 ranked guesses by the framework for experimentally observed NMR spectra of Cc1ccc(O)c(O)c1

Figure 3.12: Target Molecule and Top3 Guesses as returned by the agent along with the reward and their predicted spectra for cases when the Top1 guess is the correct guess



(a) Example 1: Top3 ranked guesses by the framework for experimentally observed NMR spectra of Cc1c(O)c(=O)oc1=O



(b) Example 2: Top3 ranked guesses by the framework for experimentally observed NMR spectra of Cc1cc(=O)ccc1=O

Figure 3.13: Target Molecule and Top3 Guesses as returned by the agent along with the reward and their predicted spectra for cases when the Top1 guess is the incorrect guess

3.4 Conclusion

This thesis provides a framework using graph convolution networks and reinforcement learning to solve the inverse molecular problem of NMR spectra. The work also introduces a novel method to train the policy and value networks *apriori* in guided MCTS runs (Training Mode on) and demonstrates the utility of Monte Carlo Tree Searches in navigating the chemical space. Unlike other prior attempts to solve this problem like the one by Jonas [85] where the model makes a prediction only 50% of the time (even though their work is tested on molecules with 32 heavy atoms), or the work by Zhang et al. [139] where the model is tested only on 9 hand-picked target spectra, this model shows good promise by predicting the correct structure among its Top3 guesses, $\sim 80\%$ of the time. Additionally, it is observed that the proposed framework performs better than brute-force checking in an enumerated database of known molecular structures. On average, it is seen that the framework calls the forward model for less than 7% of the molecules in QM9 which have the same molecular formula as the target structure, while still identifying the molecule correctly. More information on comparison of the framework against brute-force enumeration is provided in the supplementary material. Still, there are various avenues for improvement for future work. Since the RL algorithm is dependent on the forward model for its intermediate reward, a better scoring function would potentially improve the prediction accuracy. ^{13}C NMR is just one of the many spectroscopy techniques that are widely used. For example, ^{1}H NMR spectroscopy has a higher signal to noise ratio owing to the significantly larger abundance of spin-active isotope. Infrared spectroscopy sheds light on the vibrational transitions in a molecule and is considered to be complementary to NMR spectroscopy for characterizing small organic molecules. A promising extension of work presented in this thesis would be to incorporate other spectral data and leverage different kinds of information to elucidate the correct structure of an unknown molecule. Finally, we believe that proposed work is a crucial step in high-throughput synthesis, where swift and efficient verification of structures generated can make the whole process of drug discovery more robust and reliable.

Appendix A

Important sets of work discussed in relation to the inverse problems discussed in Chapter 2

Table A.1: Representative list of publications that have proposed methodologies related to molecular generation

Title	Methods	Authors
Automatic chemical design using a data-driven continuous representation of molecules	VAE with RNN Encoder and Decoder	Gómez-Bombarelli et al. [35]
Junction tree variational autoencoder for molecular graph generation	VAE with Graph Encoder and Decoder	Jin et al. [40]
GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders	VAE with Graph Encoder and Decoder	Simonovsky and Komodakis [36]
Constrained graph variational autoencoders for molecule design	VAE with Gated Graph NN Encoder and Decoder	Liu et al. [39]
Molecular generative model based on conditional variational autoencoder for de novo molecular design	VAE with RNN Encoder and Decoder	Lim et al. [163]
Grammar variational autoencoder	VAE with RNN Encoder and Decoder	Kusner et al. [164]
Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations	VAE with RNN/CNN Encoder and RNN Decoder	Winter et al. [165]

Table A.1: Representative list of publications that have proposed methodologies related to molecular generation

Title	Methods	Authors
Constrained Bayesian optimization for automatic chemical design using variational autoencoders	VAE with RNN Encoder and Decoder with Bayesian Optimization	Griffiths and Hernández-Lobato [71]
Efficient multi-objective molecular optimization in a continuous latent space	VAE with RNN Encoder and Decoder with Particle Swarm Optimization	Winter et al. [72]
druGAN: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico	Autoencoder with adverserial training	Kadurin et al. [69]
Application of generative autoencoder in de novo molecular design	Autoencoder with adverserial training followed by Bayesian Optimization	Blaschke et al. [73]
Deep reinforcement learning for de novo drug design	RNN with Reinforcement Learning	Popova et al. [47]
MoleGuLAR: Molecule Generation using Reinforcement Learning with Alternating Reward	RNN with Reinforcement Learning	Goel et al. [22]
Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models	GAN with Reinforcement Learning	Guimaraes et al. [43]
Optimizing distributions over molecular space. An objective-reinforced generative adversarial network for inverse-design chemistry (ORGANIC)	GAN with Reinforcement Learning	Sanchez-Lengeling et al. [78]
Reinforced adversarial neural computer for de novo molecular design	GAN with Reinforcement Learning	Putin et al. [79]
Adversarial threshold neural computer for molecular de novo design	GAN with Reinforcement Learning	Putin et al. [80]
MolGAN: An implicit generative model for small molecular graphs	WGAN with Reinforcement Learning	De Cao and Kipf [81]

Table A.1: Representative list of publications that have proposed methodologies related to molecular generation

Title	Methods	Authors
Mol-CycleGAN: a generative model for molecular optimization	GAN with Reinforcement Learning	Maziarka et al. [44]
A de novo molecular generation method using latent vector based generative adversarial network	Combination of VAE and GAN	Prykhodko et al. [82]
Graph convolutional policy network for goal-directed molecular graph generation	Graph NN with Reinforcement Learning	You et al. [31]
DeepGraphMolGen, a multi-objective, computational strategy for generating molecules with desirable properties: a graph convolution and reinforcement learning approach	Graph NN with Reinforcement Learning	Khemchandani et al. [166]
Optimization of molecules via deep reinforcement learning	Graph NN with Reinforcement Learning	Zhou et al. [46]

Table A.2: Representative list of publications related to application of modern ML methods for mapping retrosynthesis pathways

Title	Methods	Authors
Efficient syntheses of diverse, medicinally relevant targets planned by computer and executed in the laboratory	Only Reaction Templates	Klucznik et al. [89]
Prediction and interpretable visualization of retrosynthetic reactions using graph convolutional networks	GCN with Reaction Templates	Ishida et al. [98]
Deep retrosynthetic reaction prediction using local reactivity and global attention	GLN with Reaction Templates	Chen and Jung [99]
Planning chemical syntheses with deep neural networks and symbolic AI	MCTS with Reaction Templates	Segler et al. [100]
Learning retrosynthetic planning through simulated experience	MCTS with Reaction Templates	Schreck et al. [101]
AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning	MCTS with Reaaction Templates	Genheden et al. [102]
Retro*: Learning Retrosynthetic Planning with Neural Guided A* Search	Best First Search with Reaction Templates	Chen et al. [103]
Retrosynthetic reaction prediction using neural sequence-to-sequence model	Template Free RNN	Liu et al. [104]
A transformer model for retrosynthesis	Template Free Transformer	Karpov et al. [106]
Predicting retrosynthetic reactions using self-corrected transformer neural networks	Template Free Transformer	Zheng et al. [107]
Valid, Plausible, and Diverse Retrosynthesis Using Tied Two-Way Transformers with Latent Variables	Template Free Transformer	Kim et al. [108]
Molecular graph enhanced transformer for retrosynthesis prediction	Template Free GNN and Transformer	Mao et al. [109]
GTA: Graph Truncated Attention for Retrosynthesis	Template Free GNN with Attention	Seo et al. [110]
Automatic retrosynthetic route planning using template-free models	Template Free Transformer with MCTS	Lin et al. [111]
Predicting retrosynthetic pathways using transformer-based models and a hypergraph exploration strategy	Template Free Transformer with Beam Search	Schwaller et al. [112]
A graph to graphs framework for retrosynthesis prediction	Template Free Graph Convolutional Network	Shi et al. [114]
Learning graph models for template-free retrosynthesis	Template Free Message Passing Network	Somnath et al. [116]
Retroxpert: Decompose retrosynthesis prediction like a chemist	Semi-Template Based Graph Attention Network	Yan et al. [118]

Table A.3: ML based studies that attempted to decipher spectra to molecule inverse problem

Title	Methods	Authors
Chiral Molecular Structure Determination for a Desired Compound Just from Its Molecular Formula and Vibrational Optical Activity Spectra	SVM	Wang et al. [137]
Spectral deep learning for prediction and prospective validation of functional groups	MLP	Fine et al. [138]
NMR-TS: de novo molecule identification from NMR spectra	Tree Search, DFT Calculations	Zhang et al. [139]
Deep imitation learning for molecular inverse problems	MLP, GCN	Jonas [85]
Spectra To Structure : Deep Reinforcement Learning for Molecular Inverse Problem	MCTS, GCN	Sridharan et al. [140]

Appendix B

Supplementary Information To Chapter 3

Table 1 : Crossvalidation Accuracy

The total dataset of 2134 molecules was randomly split into 5 equal groups. In each of the five experiments, one of the groups was chosen as the hold-out test dataset and the model was trained on the remaining four groups. For each molecule, the agent made a number of guesses depending on how many episodes it ran. There were 20 processes initiated, with each of them running an episode. The accuracy of the 5 experiments ran are given in the table below:

Table B.1: Crossvalidation accuracy for 5 different folds

	set1	set2	set3	set4	set5
Top1	56.7%	56.2%	57.9%	58.3%	56.8%
Top3	81.0%	79.2%	78.4%	78.0%	79.8%
Top5	87.1%	86.4%	84.7%	85.9%	84.7%
Top7	89.7%	88.5%	87.6%	89.0%	86.7%
All Guesses	94.8%	93.9%	92.5%	92.0%	95.8%

Table 2 : Dependence of Accuracy on $nmcts$

$nmcts$ is the number of times that the search tree is traversed from the root to leaf node to explore different branches before making a true action in the current state.The dataset was divided in train-test split of ratio 90-10 and trained. During the test time, the agent was ran with varying value of the variable $nmcts$ for its MCTS Searches. The accuracy of each of these experiment is shown in the table below:

Table B.2: $nmcts$ vs Accuracy

$ncmts \longrightarrow$	50	100	200	400	1000
Top1	59.8%	57.6%	59.8%	55.4%	55.2%
Top3	71.9%	76.8%	78.1%	78.1%	77.1%
Top5	75%	80.1%	83.5%	83.9%	83.3%
Top7	77.2%	83.9%	87.1%	87.5%	87.1%
All Guesses	77.2%	85.3%	89.7%	92.9%	94.8%

Table 3 : Holdout testing by training on molecules with number of atoms < 7 and testing on molecules with number of atoms < 10

In another experiment, molecules with < 7 non-hydrogen atoms were filtered from the dataset. After running the agent on these filtered set of molecules with training mode on, the agent was tested on 200 randomly sampled molecules with ≥ 7 and < 10 heavy atoms. The accuracy of the agent on the test in this experiment is shown in the table below:

Table B.3: Accuracy in holdout testing

	Accuracy
Top1	48.5%
Top3	72.5%
Top5	78.0%
Top7	80.5%
All Guesses	86.5%

Table 4 : Comparison with forward model operated on enumerated QM9 dataset

The first column in the above table contains the SMILES of the target molecule for a few randomly chosen examples. The second column shows the number of molecules in the QM9 database with the same molecular formula as the target structure. An endeavour that tries to search for the correct structure of spectra by ranking an enumerated database would have to rank all these candidate molecules using the forward model. The third column shows the number of times the framework discussed in the manuscript calls for the forward model while resolving the unknown spectra. We note that less than about 7% of all the molecules in QM9 have the forward model called on them on average, yet the correct molecular structure was found by the framework.

It is observed that even after enumerating the chemical space, only a part of the problem is solved. For instance, the spectral distance between the experimentally observed spectra and the one predicted by the forward model is sometimes not the ideal way of ranking potential candidates, as discussed

Table B.4: Comparing forward calls on a bruteforce method

Target Smiles	**Total Mols in QM9 with the same molecular formula as target**	**Number of Forward calls made by the framework**
COC(=O)C(C)=O	915	21
CCOC(=O)CC(C)=O	7082	73
COC(CCC#N)OC	14329	155
C1=CCOCOC1	498	218
Cc1ccc(O)c(O)c1	16325	172
CCCCC=CC(C)=O	13572	169
CC=CCCCO	570	64
O=C(O)C1CC(O)CN1	3916	510
CN(C)c1nccnn1	1128	341

in the manuscript. Often, it is because of the inaccuracy of the forward model that the correct target molecules are ranked lower. This problem would only get enhanced when the entire QM9 database is ranked against each other. For example, more than 7,000 molecules have the same molecular formula as the target molecule CCOC(=O)CC(C)=O in the qm9 database; ranking all of them using the forward model is potentially expensive and time-consuming. On the other hand, the framework mentioned in this manuscript makes only 73 forward prediction calls in the process of correctly identifying the target structure. Another consequence of the above is that the rank of the correct structure when ranking against the 7000 molecules in the qm9 database is 13th where as the correct structure is ranked 1st among the candidate structures as returned by our framework.

Similarly, in another example, the molecule CC(=O)N1CC=CC1=O has more than 14,000 molecules with the same molecular formula in qm9. Ranking all of them takes 10m 12s, with the target molecule being at the rank of 43 after ranking using the scoring reward. Whereas the framework in the manuscript takes 105 seconds to complete all of its runs, the target molecule is at the rank of 7 among the output candidate structures.

This is again indicative of the model's ability to interpret spectral data and navigate the chemical space efficiently and return only the most promising candidate structures.

Related Publications

1. **Bhuvanesh Sridharan**, Sarvesh Mehta, Yashaswi Pathak, and U. Deva Priyakumar. "Deep Reinforcement Learning for Molecular Inverse Problem of Nuclear Magnetic Resonance Spectra to Molecular Structure." The Journal of Physical Chemistry Letters 13, no. 22 (2022): 4924-4933.

2. **Bhuvanesh Sridharan***, Manan Goel*, and U. Deva Priyakumar. "Modern machine learning for tackling inverse problems in chemistry: molecular design to realization." Chemical Communications 58, no. 35 (2022): 5316-5331.

3. Manan Goel*, Rishal Aggarwal*, **Bhuvanesh Sridharan***, Pradeep Kumar Pal, and U. Deva Priyakumar. "Efficient and enhanced sampling of drug-like chemical space for virtual screening and molecular design using modern machine learning methods." Wiley Interdisciplinary Reviews: Computational Molecular Science (2022): e1637.

4. Korlepara, Divya B., C. S. Vasavi, Shruti Jeurkar, Pradeep Kumar Pal, Subhajit Roy, Sarvesh Mehta, Shubham Sharma, Vishal Kumar, Charuvaka Muvva, **Bhuvanesh Sridharan**, Akshit-Garg, Rohit Modee, Agastya P. Bhati, Divya Nayar, and U. Deva Priyakumar. "Plas-5k: Dataset of protein-ligand affinities from molecular dynamics for machine learning applications." Scientific Data 9, no. 1 (2022): 548.

5. Sriram Devata*, **Bhuvanesh Sridharan***, Sarvesh Mehta*, and U. Deva Priyakumar. "Deep-SPInN - multimodal Deep learning for molecular Structure Prediction from Infrared and NMR spectra" [SUBMITTED]

6. **Bhuvanesh Sridharan***, Animesh Sinha*, Jai Bardhan, Rohit modee, and U. Deva Priyakumar. "Review of Reinforcement Learning in Chemistry" [SUBMITTED]

Bibliography

[1] Steven M. Paul, Daniel S. Mytelka, Christopher T. Dunwiddie, Charles C. Persinger, Bernard H. Munos, Stacy R. Lindborg, and Aaron L. Schacht. How to improve r&d productivity: the pharmaceutical industry's grand challenge. *Nature Reviews Drug Discovery*, 9(3):203–214, Mar 2010. ISSN 1474-1784. doi: 10.1038/nrd3078. URL `https://doi.org/10.1038/nrd3078`.

[2] Lars Ruddigkeit, Ruud Van Deursen, Lorenz C Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database gdb-17. *Journal of chemical information and modeling*, 52(11):2864–2875, 2012.

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

[4] Teague Sterling and John J Irwin. Zinc 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11):2324–2337, 2015.

[5] Andreas Mayr, Günter Klambauer, Thomas Unterthiner, and Sepp Hochreiter. Deeptox: toxicity prediction using deep learning. *Frontiers in Environmental Science*, 3:80, 2016.

[6] Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. Chembl: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(D1):D1100–D1107, 2012.

[7] Daniel Mark Lowe. *Extraction of chemical structures and reactions from the literature*. PhD thesis, University of Cambridge, 2012.

[8] Felix Strieth-Kalthoff, Frederik Sandfort, Marwin HS Segler, and Frank Glorius. Machine learning the ropes: principles, applications and directions in synthetic chemistry. *Chemical Society Reviews*, 49(17):6154–6168, 2020.

[9] Keith T Butler, Daniel W Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. Machine learning for molecular and materials science. *Nature*, 559(7715):547–555, 2018.

[10] Charles W Groetsch and CW Groetsch. *Inverse problems in the mathematical sciences*, volume 52. Springer, 1993.

[11] Francisco Duarte Moura Neto and Antônio José da Silva Neto. *An introduction to inverse problems with applications*. Springer Science & Business Media, 2012.

[12] James L Buchanan, Robert P Gilbert, Armand Wirgin, and Yongzhi S Xu. *Marine acoustics: direct and inverse problems*. SIAM, 2004.

[13] Heinz W Engl, Alfred K Louis, and William Rundell. *Inverse Problems in Medical Imaging and Nondestructive Testing: Proceedings of the Conference in Oberwolfach, Federal Republic of Germany, February 4–10, 1996*. Springer Science & Business Media, 2012.

[14] J Carlos Santamarina and Dante Fratta. *Discrete signals and inverse problems: an introduction for engineers and scientists*. John Wiley & Sons, 2005.

[15] Andrew Goldenberg, Beno Benhabib, and Robert Fenton. A complete generalized solution to the inverse kinematics of robots. *IEEE Journal on Robotics and Automation*, 1(1):14–20, 1985.

[16] Jacek Karwowski. Inverse problems in quantum chemistry. *International Journal of Quantum Chemistry*, 109(11):2456–2463, 2009.

[17] Xueliang Li, Zimao Li, and Lusheng Wang. The inverse problems for some topological indices in combinatorial chemistry. *Journal of Computational Biology*, 10(1):47–55, 2003.

[18] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.

[19] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[20] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[21] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[22] Manan Goel, Shampa Raghunathan, Siddhartha Laghuvarapu, and U Deva Priyakumar. Molegular: Molecule generation using reinforcement learning with alternating rewards. 2021.

[23] Viraj Bagal, Rishal Aggarwal, PK Vinod, and U Deva Priyakumar. Molgpt: Molecular generation using a transformer-decoder model. *Journal of Chemical Information and Modeling*, 2021.

[24] Xiufeng Yang, Jinzhe Zhang, Kazuki Yoshizoe, Kei Terayama, and Koji Tsuda. Chemts: an efficient python library for de novo molecular generation. *Science and technology of advanced materials*, 18(1):972–976, 2017.

[25] Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1): 120–131, 2018.

[26] Daniel Neil, Marwin Segler, Laura Guasch, Mohamed Ahmed, Dean Plumbley, Matthew Sellwood, and Nathan Brown. Exploring deep recurrent models with reinforcement learning for molecule design. 2018.

[27] Benjamin Sanchez-Lengeling, Emily Reif, Adam Pearce, and Alexander B Wiltschko. A gentle introduction to graph neural networks. *Distill*, 6(9):e33, 2021.

[28] Oliver Wieder, Stefan Kohlbacher, Mélaine Kuenemann, Arthur Garon, Pierre Ducrot, Thomas Seidel, and Thierry Langer. A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies*, 2020.

[29] Yashaswi Pathak, Sarvesh Mehta, and U Deva Priyakumar. Learning atomic interactions through solvation free energy prediction using graph neural networks. *Journal of Chemical Information and Modeling*, 61(2):689–698, 2021.

[30] Jiacheng Xiong, Zhaoping Xiong, Kaixian Chen, Hualiang Jiang, and Mingyue Zheng. Graph neural networks for automated de novo drug design. *Drug Discovery Today*, 2021.

[31] Jiaxuan You, Bowen Liu, Rex Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *arXiv preprint arXiv:1806.02473*, 2018.

[32] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *arXiv preprint arXiv:2003.05991*, 2020.

[33] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[34] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL `http://arxiv.org/abs/1312.6114`.

[35] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[36] Martin Simonovsky and Nikos Komodakis. Graphvae: Towards generation of small graphs using variational autoencoders. In *International conference on artificial neural networks*, pages 412–422. Springer, 2018.

[37] Jaechang Lim, Seongok Ryu, Jin Woo Kim, and Woo Youn Kim. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *Journal of cheminformatics*, 10(1):1–9, 2018.

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

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

[40] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pages 2323–2332. PMLR, 2018.

[41] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Commun. ACM*, 63(11): 139–144, oct 2020. ISSN 0001-0782. doi: 10.1145/3422622. URL `https://doi.org/10.1145/3422622`.

[42] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[43] Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843*, 2017.

[44] Łukasz Maziarka, Agnieszka Pocha, Jan Kaczmarczyk, Krzysztof Rataj, Tomasz Danel, and Michał Warchoł. Mol-cyclegan: a generative model for molecular optimization. *Journal of Cheminformatics*, 12(1):1–18, 2020.

[45] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[46] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):1–10, 2019.

[47] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7):eaap7885, 2018.

[48] Zhenpeng Zhou, Xiaocheng Li, and Richard N Zare. Optimizing chemical reactions with deep reinforcement learning. *ACS central science*, 3(12):1337–1344, 2017.

[49] Youngwoo Cho, Sookyung Kim, Peggy Pk Li, Mike P Surh, T Yong-Jin Han, and Jaegul Choo. Physics-guided reinforcement learning for 3d molecular structures. In *Workshop at the 33rd Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[50] Kabir Ahuja, William H Green, and Yi-Pei Li. Learning to optimize molecular geometries using reinforcement learning. *Journal of Chemical Theory and Computation*, 17(2):818–825, 2021.

[51] P. G. Polishchuk, T. I. Madzhidov, and A. Varnek. Estimation of the size of drug-like chemical space based on gdb-17 data. *Journal of Computer-Aided Molecular Design*, 27(8):675–679, Aug 2013. ISSN 1573-4951. doi: 10.1007/s10822-013-9672-4. URL https://doi.org/10.1007/s10822-013-9672-4.

[52] Ruiwu Liu, Xiaocen Li, and Kit S Lam. Combinatorial chemistry in drug discovery. *Current Opinion in Chemical Biology*, 38:117–126, 2017. ISSN 1367-5931. doi: https://doi.org/10.1016/j.cbpa.2017.03.017. URL https://www.sciencedirect.com/science/article/pii/S1367593117300534. Next Generation Therapeutics.

[53] Maximilian Benz, Mijanur R Molla, Alexander Böser, Alisa Rosenfeld, and Pavel A Levkin. Marrying chemistry with biology by combining on-chip solution-based combinatorial synthesis and cellular screening. *Nature communications*, 10(1):1–10, 2019.

[54] Oleksandr O Grygorenko, Dmitriy M Volochnyuk, Sergey V Ryabukhin, and Duncan B Judd. The symbiotic relationship between drug discovery and organic chemistry. *Chemistry–A European Journal*, 26(6):1196–1237, 2020.

[55] Priska Frei, Rachel Hevey, and Beat Ernst. Dynamic combinatorial chemistry: a new methodology comes of age. *Chemistry–A European Journal*, 25(1):60–73, 2019.

[56] Sabrina Jaeger, Simone Fulle, and Samo Turk. Mol2vec: unsupervised machine learning approach with chemical intuition. *Journal of chemical information and modeling*, 58(1):27–35, 2018.

[57] Sarvesh Mehta, Siddhartha Laghuvarapu, Yashaswi Pathak, Aaftaab Sethi, Mallika Alvala, and U Deva Priyakumar. Memes: Machine learning framework for enhanced molecular screening. *Chemical science*, 12(35):11710–11721, 2021.

[58] Rishal Aggarwal, Akash Gupta, Vineeth Chelur, CV Jawahar, and U Deva Priyakumar. Deeppocket: Ligand binding site detection and segmentation using 3d convolutional neural networks. 2021.

[59] Oliver Kramer. Genetic algorithms. In *Genetic algorithm essentials*, pages 11–19. Springer, 2017.

[60] Lawrence Davis. Handbook of genetic algorithms. 1991.

[61] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning, 2017.

[62] Naruki Yoshikawa, Kei Terayama, Masato Sumita, Teruki Homma, Kenta Oono, and Koji Tsuda. Population-based de novo molecule generation, using grammatical evolution. *Chemistry Letters*, 47(11):1431–1434, 2018.

[63] Jan H Jensen. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572, 2019.

[64] Wei Ren Tan, Chee Seng Chan, Hernán E Aguirre, and Kiyoshi Tanaka. Artgan: Artwork synthesis with conditional categorical gans. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3760–3764. IEEE, 2017.

[65] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[66] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[67] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[68] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28: 3483–3491, 2015.

[69] Artur Kadurin, Sergey Nikolenko, Kuzma Khrabrov, Alex Aliper, and Alex Zhavoronkov. drugan: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Molecular pharmaceutics*, 14(9):3098–3104, 2017.

[70] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[71] Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. Constrained bayesian optimization for automatic chemical design using variational autoencoders. *Chemical science*, 11(2):577–586, 2020.

[72] Robin Winter, Floriane Montanari, Andreas Steffen, Hans Briem, Frank Noé, and Djork-Arné Clevert. Efficient multi-objective molecular optimization in a continuous latent space. *Chemical science*, 10(34):8016–8024, 2019.

[73] Thomas Blaschke, Marcus Olivecrona, Ola Engkvist, Jürgen Bajorath, and Hongming Chen. Application of generative autoencoder in de novo molecular design. *Molecular informatics*, 37 (1-2):1700123, 2018.

[74] Sowmya Ramaswamy Krishnan, Navneet Bung, Gopalakrishnan Bulusu, and Arijit Roy. Accelerating de novo drug design against novel proteins using deep learning. *Journal of Chemical Information and Modeling*, 61(2):621–630, 2021.

[75] Navneet Bung, Sowmya R Krishnan, Gopalakrishnan Bulusu, and Arijit Roy. De novo design of new chemical entities for sars-cov-2 using artificial intelligence. *Future medicinal chemistry*, 13 (06):575–585, 2021.

[76] Tirtharaj Dash, Ashwin Srinivasan, Lovekesh Vig, and Arijit Roy. Using domain-knowledge to assist lead discovery in early-stage drug design. *bioRxiv*, 2021.

[77] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[78] Benjamin Sanchez-Lengeling, Carlos Outeiral, Gabriel L Guimaraes, and Alan Aspuru-Guzik. Optimizing distributions over molecular space. an objective-reinforced generative adversarial network for inverse-design chemistry (organic). 2017.

[79] Evgeny Putin, Arip Asadulaev, Yan Ivanenkov, Vladimir Aladinskiy, Benjamin Sanchez-Lengeling, Alán Aspuru-Guzik, and Alex Zhavoronkov. Reinforced adversarial neural computer for de novo molecular design. *Journal of chemical information and modeling*, 58(6):1194–1204, 2018.

[80] Evgeny Putin, Arip Asadulaev, Quentin Vanhaelen, Yan Ivanenkov, Anastasia V Aladinskaya, Alex Aliper, and Alex Zhavoronkov. Adversarial threshold neural computer for molecular de novo design. *Molecular pharmaceutics*, 15(10):4386–4397, 2018.

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

[82] Oleksii Prykhodko, Simon Viet Johansson, Panagiotis-Christos Kotsias, Josep Arús-Pous, Esben Jannik Bjerrum, Ola Engkvist, and Hongming Chen. A de novo molecular generation method using latent vector based generative adversarial network. *Journal of Cheminformatics*, 11(1):1–13, 2019.

[83] Yash Khemchandani, Stephen O'Hagan, Soumitra Samanta, Neil Swainston, Timothy J Roberts, Danushka Bollegala, and Douglas B Kell. Deepgraphmolgen, a multi-objective, computational strategy for generating molecules with desirable properties: a graph convolution and reinforcement learning approach. *Journal of cheminformatics*, 12(1):1–17, 2020.

[84] Laurianne David, Amol Thakkar, Rocío Mercado, and Ola Engkvist. Molecular representations in ai-driven drug discovery: a review and practical guide. *Journal of Cheminformatics*, 12(1): 1–22, 2020.

[85] Eric Jonas. Deep imitation learning for molecular inverse problems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/b0bef4c9a6e50d43880191492d4fc827-Paper.pdf.

[86] Elias James Corey. *The logic of chemical synthesis*. 1991.

[87] Marc A. Shampo, Robert A. Kyle, and David P. Steensma. Elias james corey—nobel prize for retrosynthetic analysis. *Mayo Clinic Proceedings*, 88(1):e7, 2013. ISSN 0025-6196. doi: https://doi.org/10.1016/j.mayocp.2012.01.024. URL https://www.sciencedirect.com/science/article/pii/S0025619612010427.

[88] Bartosz A Grzybowski, Sara Szymkuć, Ewa P Gajewska, Karol Molga, Piotr Dittwald, Agnieszka Wołos, and Tomasz Klucznik. Chematica: a story of computer code that started to think like a chemist. *Chem*, 4(3):390–398, 2018.

[89] Tomasz Klucznik, Barbara Mikulak-Klucznik, Michael P McCormack, Heather Lima, Sara Szymkuć, Manishabrata Bhowmick, Karol Molga, Yubai Zhou, Lindsey Rickershauser, Ewa P Gajewska, et al. Efficient syntheses of diverse, medicinally relevant targets planned by computer and executed in the laboratory. *Chem*, 4(3):522–532, 2018.

[90] Yuning Shen, Julia E Borowski, Melissa A Hardy, Richmond Sarpong, Abigail G Doyle, and Tim Cernak. Automation and computer-assisted planning for chemical synthesis. *Nature Reviews Methods Primers*, 1(1):1–23, 2021.

[91] Connor W Coley, William H Green, and Klavs F Jensen. Machine learning in computer-aided synthesis planning. *Accounts of chemical research*, 51(5):1281–1289, 2018.

[92] Daniel Mark Lowe. *Extraction of chemical structures and reactions from the literature*. PhD thesis, University of Cambridge, 2012.

[93] Pieter P Plehiers, Guy B Marin, Christian V Stevens, and Kevin M Van Geem. Automated reaction database and reaction network analysis: extraction of reaction templates using cheminformatics. *Journal of cheminformatics*, 10(1):1–18, 2018.

[94] Syed Asad Rahman, Gilliean Torrance, Lorenzo Baldacci, Sergio Martínez Cuesta, Franz Fenninger, Nimish Gopal, Saket Choudhary, John W May, Gemma L Holliday, Christoph Steinbeck, et al. Reaction decoder tool (rdt): extracting features from chemical reactions. *Bioinformatics*, 32(13):2065–2066, 2016.

[95] Connor W Coley, William H Green, and Klavs F Jensen. Rdchiral: An rdkit wrapper for handling stereochemistry in retrosynthetic template extraction and application. *Journal of chemical information and modeling*, 59(6):2529–2537, 2019.

[96] James Law, Zsolt Zsoldos, Aniko Simon, Darryl Reid, Yang Liu, Sing Yoong Khew, A Peter Johnson, Sarah Major, Robert A Wade, and Howard Y Ando. Route designer: a retrosynthetic analysis tool utilizing automated retrosynthetic rule generation. *Journal of chemical information and modeling*, 49(3):593–602, 2009.

[97] Siqi Hong, Hankz Hankui Zhuo, Kebing Jin, and Zhanwen Zhou. Retrosynthetic planning with experience-guided monte carlo tree search. *arXiv preprint arXiv:2112.06028*, 2021.

[98] Shoichi Ishida, Kei Terayama, Ryosuke Kojima, Kiyosei Takasu, and Yasushi Okuno. Prediction and interpretable visualization of retrosynthetic reactions using graph convolutional networks. *Journal of chemical information and modeling*, 59(12):5026–5033, 2019.

[99] Shuan Chen and Yousung Jung. Deep retrosynthetic reaction prediction using local reactivity and global attention. *JACS Au*, 1(10):1612–1620, 2021.

[100] Marwin HS Segler, Mike Preuss, and Mark P Waller. Planning chemical syntheses with deep neural networks and symbolic ai. *Nature*, 555(7698):604–610, 2018.

[101] John S Schreck, Connor W Coley, and Kyle JM Bishop. Learning retrosynthetic planning through simulated experience. *ACS central science*, 5(6):970–981, 2019.

[102] Samuel Genheden, Amol Thakkar, Veronika Chadimová, Jean-Louis Reymond, Ola Engkvist, and Esben Bjerrum. Aizynthfinder: a fast, robust and flexible open-source software for retrosynthetic planning. *Journal of cheminformatics*, 12(1):1–9, 2020.

[103] Binghong Chen, Chengtao Li, Hanjun Dai, and Le Song. Retro*: Learning retrosynthetic planning with neural guided a* search. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1608–1616. PMLR, 2020. URL `http://proceedings.mlr.press/v119/chen20k.html`.

[104] Bowen Liu, Bharath Ramsundar, Prasad Kawthekar, Jade Shi, Joseph Gomes, Quang Luu Nguyen, Stephen Ho, Jack Sloane, Paul Wender, and Vijay Pande. Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ACS central science*, 3(10):1103–1113, 2017.

[105] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[106] Pavel Karpov, Guillaume Godin, and Igor V Tetko. A transformer model for retrosynthesis. In *International Conference on Artificial Neural Networks*, pages 817–830. Springer, 2019.

[107] Shuangjia Zheng, Jiahua Rao, Zhongyue Zhang, Jun Xu, and Yuedong Yang. Predicting retrosynthetic reactions using self-corrected transformer neural networks. *Journal of Chemical Information and Modeling*, 60(1):47–55, 2019.

[108] Eunji Kim, Dongseon Lee, Youngchun Kwon, Min Sik Park, and Youn-Suk Choi. Valid, plausible, and diverse retrosynthesis using tied two-way transformers with latent variables. *Journal of Chemical Information and Modeling*, 61(1):123–133, 2021.

[109] Kelong Mao, Xi Xiao, Tingyang Xu, Yu Rong, Junzhou Huang, and Peilin Zhao. Molecular graph enhanced transformer for retrosynthesis prediction. *Neurocomputing*, 457:193–202, 2021.

[110] Seung-Woo Seo, You Young Song, June Yong Yang, Seohui Bae, Hankook Lee, Jinwoo Shin, Sung Ju Hwang, and Eunho Yang. Gta: Graph truncated attention for retrosynthesis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 531–539, 2021.

[111] Kangjie Lin, Youjun Xu, Jianfeng Pei, and Luhua Lai. Automatic retrosynthetic route planning using template-free models. *Chemical Science*, 11(12):3355–3364, 2020.

[112] Philippe Schwaller, Riccardo Petraglia, Valerio Zullo, Vishnu H Nair, Rico Andreas Haeuselmann, Riccardo Pisoni, Costas Bekas, Anna Iuliano, and Teodoro Laino. Predicting retrosynthetic pathways using transformer-based models and a hyper-graph exploration strategy. *Chemical science*, 11(12):3316–3325, 2020.

[113] Connor W Coley, Luke Rogers, William H Green, and Klavs F Jensen. Scscore: synthetic complexity learned from a reaction corpus. *Journal of chemical information and modeling*, 58(2): 252–261, 2018.

[114] Chence Shi, Minkai Xu, Hongyu Guo, Ming Zhang, and Jian Tang. A graph to graphs framework for retrosynthesis prediction. In *International Conference on Machine Learning*, pages 8818–8827. PMLR, 2020.

[115] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

[116] Vignesh Ram Somnath, Charlotte Bunne, Connor W Coley, Andreas Krause, and Regina Barzilay. Learning graph models for template-free retrosynthesis. *arXiv preprint arXiv:2006.07038*, 2020.

[117] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[118] Chaochao Yan, Qianggang Ding, Peilin Zhao, Shuangjia Zheng, Jinyu Yang, Yang Yu, and Junzhou Huang. Retroxpert: Decompose retrosynthesis prediction like a chemist. *arXiv preprint arXiv:2011.02893*, 2020.

[119] Daniela S Mattes, Nicole Jung, Laura K Weber, Stefan Bräse, and Frank Breitling. Miniaturized and automated synthesis of biomolecules—overview and perspectives. *Advanced Materials*, 31 (26):1806656, 2019.

[120] Melanie Trobe and Martin D Burke. The molecular industrial revolution: automated synthesis of small molecules. *Angewandte Chemie International Edition*, 57(16):4192–4214, 2018.

[121] Hanyu Gao, Thomas J Struble, Connor W Coley, Yuran Wang, William H Green, and Klavs F Jensen. Using machine learning to predict suitable conditions for organic reactions. *ACS central science*, 4(11):1465–1476, 2018.

[122] Alain C Vaucher, Federico Zipoli, Joppe Geluykens, Vishnu H Nair, Philippe Schwaller, and Teodoro Laino. Automated extraction of chemical synthesis actions from experimental procedures. *Nature communications*, 11(1):1–11, 2020.

[123] Junqi Li, Steven G Ballmer, Eric P Gillis, Seiko Fujii, Michael J Schmidt, Andrea ME Palazzolo, Jonathan W Lehmann, Greg F Morehouse, and Martin D Burke. Synthesis of many different types of organic small molecules using one automated process. *Science*, 347(6227):1221–1226, 2015.

[124] Sebastian Steiner, Jakob Wolf, Stefan Glatzel, Anna Andreou, Jarosław M Granda, Graham Keenan, Trevor Hinkley, Gerardo Aragon-Camarasa, Philip J Kitson, Davide Angelone, et al. Organic synthesis in a modular robotic system driven by a chemical programming language. *Science*, 363(6423):10, 2019.

[125] Piotr S Gromski, Jarosław M Granda, and Leroy Cronin. Universal chemical synthesis and discovery with 'the chemputer'. *Trends in Chemistry*, 2(1):4–12, 2020.

[126] Nathan Collins, David Stout, Jin-Ping Lim, Jeremiah P Malerich, Jason D White, Peter B Madrid, Mario Latendresse, David Krieger, Judy Szeto, Vi-Anh Vu, et al. Fully automated chemical synthesis: toward the universal synthesizer. *Organic Process Research & Development*, 24(10): 2064–2077, 2020.

[127] Connor W Coley, Dale A Thomas, Justin AM Lummiss, Jonathan N Jaworski, Christopher P Breen, Victor Schultz, Travis Hart, Joshua S Fishman, Luke Rogers, Hanyu Gao, et al. A robotic platform for flow synthesis of organic compounds informed by ai planning. *Science*, 365(6453): 10, 2019.

[128] Jarosław M Granda, Liva Donina, Vincenza Dragone, De-Liang Long, and Leroy Cronin. Controlling an organic synthesis robot with machine learning to search for new reactivity. *Nature*, 559(7714):377–381, 2018.

[129] Benjamin J Shields, Jason Stevens, Jun Li, Marvin Parasram, Farhan Damani, Jesus I Martinez Alvarado, Jacob M Janey, Ryan P Adams, and Abigail G Doyle. Bayesian reaction optimization as a tool for chemical synthesis. *Nature*, 590(7844):89–96, 2021.

[130] Loïc M Roch, Florian Häse, Christoph Kreisbeck, Teresa Tamayo-Mendoza, Lars PE Yunker, Jason E Hein, and Alán Aspuru-Guzik. Chemos: An orchestration software to democratize autonomous discovery. *PLoS One*, 15(4):e0229862, 2020.

[131] Ian M Pendleton, Gary Cattabriga, Zhi Li, Mansoor Ani Najeeb, Sorelle A Friedler, Alexander J Norquist, Emory M Chan, and Joshua Schrier. Experiment specification, capture and laboratory automation technology (escalate): a software pipeline for automated chemical experimentation and data management. *MRS Communications*, 9(3):846–859, 2019.

[132] Katharine Sanderson. Automation: Chemistry shoots for the moon. *Nature*, 568(7752):577–580, 2019.

[133] Shungo Koichi, Masaki Arisaka, Hiroyuki Koshino, Atsushi Aoki, Satoru Iwata, Takeaki Uno, and Hiroko Satoh. Chemical structure elucidation from 13c nmr chemical shifts: Efficient data processing using bipartite matching and maximal clique algorithms. *Journal of chemical information and modeling*, 54(4):1027–1035, 2014.

[134] JFG Vliegenthart, JA van Kuik, and K Hård. A 1h nmr database computer program for the analysis of the primary structure of complex carbohydrates. *Carbohydrate research*, 235:53–68, 1992.

[135] Fingerprint region, aug 24 2020. URL `https://chem.libretexts.org/@go/page/40288`.

[136] Barbara H. Stuart. *Infrared Spectroscopy: Fundamentals and Applications*. In Stuart [136], 2004. ISBN 9780470854273. doi: 10.1002/0470011149.

[137] Zhimeng Wang, Xiaoyu Feng, Junhong Liu, Minchun Lu, and Menglong Li. Functional groups prediction from infrared spectra based on computer-assist approaches. *Microchemical Journal*, 159:105395, 2020. ISSN 0026-265X. doi: https://doi.org/10.1016/j.microc.2020.105395. URL `https://www.sciencedirect.com/science/article/pii/S0026265X20320853`.

[138] Jonathan A. Fine, Anand A. Rajasekar, Krupal P. Jethava, and Gaurav Chopra. Spectral deep learning for prediction and prospective validation of functional groups. *Chem. Sci.*, 11:4618–4630, 2020. doi: 10.1039/C9SC06240H. URL `http://dx.doi.org/10.1039/C9SC06240H`.

[139] Jinzhe Zhang, Kei Terayama, Masato Sumita, Kazuki Yoshizoe, Kengo Ito, Jun Kikuchi, and Koji Tsuda. Nmr-ts: de novo molecule identification from nmr spectra. *Science and Technology of Advanced Materials*, 21(1):552–561, 2020. doi: 10.1080/14686996.2020.1793382. URL https://doi.org/10.1080/14686996.2020.1793382. PMID: 32939179.

[140] Bhuvanesh Sridharan, Sarvesh Mehta, Yashaswi Pathak, and U Deva Priyakumar. Spectra to structure: Deep reinforcement learning for molecular inverse problem. 2021.

[141] Hongchao Ji, Hanzi Deng, Hongmei Lu, and Zhimin Zhang. Predicting a molecular fingerprint from an electron ionization mass spectrum with deep neural networks. *Analytical Chemistry*, 92 (13):8649–8653, 2020. doi: 10.1021/acs.analchem.0c01450. URL https://doi.org/10.1021/acs.analchem.0c01450. PMID: 32584545.

[142] Youzhong Liu, Aida Mrzic, Pieter Meysman, Thomas De Vijlder, Edwin P. Romijn, Dirk Valkenborg, Wout Bittremieux, and Kris Laukens. Messar: Automated recommendation of metabolite substructures from tandem mass spectra. *PLOS ONE*, 15(1):1–17, 01 2020. doi: 10.1371/journal.pone.0226770. URL https://doi.org/10.1371/journal.pone.0226770.

[143] Eleni Litsa, Vijil Chenthamarakshan, Payel Das, and Lydia Kavraki. Spec2mol: An end-to-end deep learning framework for translating ms/ms spectra to de-novo molecules. *ChemRxiv*, 2021. doi: 10.33774/chemrxiv-2021-6rdh6.

[144] Erico Guizzo. Robots with their heads in the clouds. *IEEE Spectrum*, 48(3):16–18, 2011.

[145] Benjamin Pitzer, Sarah Osentoski, Philip Roan, C Bersh, and Jan Becker. Making robots cheaper, more capable, and safer. In *The PR2 Workshop: Results, Challenges and Lessons Learned in Advancing Robots with a Common Platform, IROS*. Citeseer, 2011.

[146] Sam Asami, Peter Schmieder, and Bernd Reif. High resolution 1h-detected solid-state nmr spectroscopy of protein aliphatic resonances: Access to tertiary structure information. *J. Am. Chem. Soc.*, 132(43):15133–15135, Nov 2010. ISSN 0002-7863. doi: 10.1021/ja106170h. URL https://doi.org/10.1021/ja106170h.

[147] Christian A.E.M. Spronk, Jens P. Linge, Cornelis W. Hilbers, and Geerten W. Vuister. Improving the quality of protein structures derived by nmr spectroscopy**. *J. Biomol. NMR*, 22(3):281–289, Mar 2002. ISSN 1573-5001. doi: 10.1023/A:1014971029663. URL https://doi.org/10.1023/A:1014971029663.

[148] A. Y. S. Balazs, N. L. Davies, D. Longmire, M. J. Packer, and E. Chiarparin. Nuclear magnetic resonance free ligand conformations and atomic resolution dynamics. *Magnetic Resonance*, 2 (1):489–498, 2021. doi: 10.5194/mr-2-489-2021. URL https://mr.copernicus.org/articles/2/489/2021/.

[149] Michael W Lodewyk, Matthew R Siebert, and Dean J Tantillo. Computational prediction of 1h and 13c chemical shifts: a useful tool for natural product, mechanistic, and synthetic organic chemistry. *Chem. Rev. (Washington, DC, U. S.)*, 112(3):1839–1862, 2012.

[150] Eric Jonas and Stefan Kuhn. Rapid prediction of nmr spectral properties with quantified uncertainty. *J. Cheminf.*, 11(1):50, Aug 2019. ISSN 1758-2946. doi: 10.1186/s13321-019-0374-3. URL https://doi.org/10.1186/s13321-019-0374-3.

[151] Eric Jonas, Stefan Kuhn, and Nils Schlörer. Prediction of chemical shift in nmr: a review. *Magn. Reson. Chem.*, 2021.

[152] W. Bremser. Hose — a novel substructure code. *Anal. Chim. Acta*, 103(4):355–365, 1978. ISSN 0003-2670. doi: https://doi.org/10.1016/S0003-2670(01)83100-7. URL https://www.sciencedirect.com/science/article/pii/S0003267001831007.

[153] Brian Cherinka, Brett H. Andrews, José Sánchez-Gallego, Joel Brownstein, María Argudo-Fernández, Michael Blanton, Kevin Bundy, Amy Jones, Karen Masters, David R. Law, Kate Rowlands, Anne-Marie Weijmans, Kyle Westfall, and Renbin Yan. Marvin: A tool kit for streamlined access and visualization of the SDSS-IV MaNGA data set. *Astron. J.*, 158(2):74, jul 2019. doi: 10.3847/1538-3881/ab2634. URL https://doi.org/10.3847/1538-3881/ab2634.

[154] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy P. Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR*, abs/1712.01815, 2017. URL http://arxiv.org/abs/1712.01815.

[155] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. *CoRR*, abs/1704.01212, 2017. URL http://arxiv.org/abs/1704.01212.

[156] Stefan Kuhn and Nils E. Schlörer. Facilitating quality control for spectra assignments of small organic molecules: nmrshiftdb2 – a free in-house nmr database with integrated lims for academic service laboratories. *Magn. Reson. Chem.*, 53(8):582–589, 2015. doi: https://doi.org/10.1002/mrc.4263. URL https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/abs/10.1002/mrc.4263.

[157] Bhuvanesh Sridharan, Manan Goel, and U Deva Priyakumar. Modern machine learning for tackling inverse problems in chemistry: Molecular design to realization. *Chem. Comm.*, 2022.

[158] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *J. Artif. Intell. Res.*, 4:237–285, 1996.

[159] Thomas M. Moerland, Joost Broekens, Aske Plaat, and Catholijn M. Jonker. Monte carlo tree search for asymmetric trees, 2018. URL https://arxiv.org/abs/1805.09218.

[160] Aaditya Ramdas, Nicolas Garcia, and Marco Cuturi. On wasserstein two sample testing and related families of nonparametric tests, 2015. URL https://arxiv.org/abs/1509.02237.

[161] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Adv. Neural Inf. Process. Syst.*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/f9be311e65d81a9ad8150a60844bb94c-Paper.pdf.

[162] Greg Landrum. Rdkit: Open-source cheminformatics software. 2016. URL https://github.com/rdkit/rdkit/releases/tag/Release_2016_09_4.

[163] Jaechang Lim, Seongok Ryu, Jin Woo Kim, and Woo Youn Kim. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *Journal of cheminformatics*, 10(1):1–9, 2018.

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

[165] Robin Winter, Floriane Montanari, Frank Noé, and Djork-Arné Clevert. Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. *Chemical science*, 10(6):1692–1701, 2019.

[166] Yash Khemchandani, Stephen O'Hagan, Soumitra Samanta, Neil Swainston, Timothy J Roberts, Danushka Bollegala, and Douglas B Kell. Deepgraphmolgen, a multi-objective, computational strategy for generating molecules with desirable properties: a graph convolution and reinforcement learning approach. *Journal of cheminformatics*, 12(1):1–17, 2020.