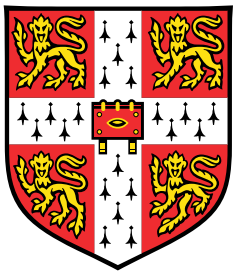
Automatic Chemical Design with Molecular Graph Variational Autoencoders



Richard Devin Shen

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy

Homerton College

August 2018

Declaration

I, Richard Shen of Homerton College, being a candidate for the M.Phil in Machine Learning, Speech and Language Technology, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose. This dissertation contains 13,150 words.

Richard Devin Shen
August 2018

I would like to dedicate this thesis to my endlessly patient and ever supportive parents.
I will never be able to say it enough, but I hope this message makes a modicum of progress
in expressing how much I love you and appreciate all the sacrifices you have made for me.

Acknowledgements

I would like to thank my supervisor, Dr. José Miguel Hernández-Lobato, for his guidance in setting the scope of this project and for his invaluable feedback in specifying particular interesting directions for further study and experimentation.

Abstract

Traditional, more biochemically motivated approaches to chemical design and drug discovery are notoriously complex and costly processes. The space of all synthesizable molecules is far too large to exhaustively search any meaningful subset for interesting novel drug and molecule proposals, and the lack of any particularly informative and manageable structure to this search space makes the very task of defining interesting subsets a difficult problem in itself. Recent years have seen the proposal and rapid development of alternative, machine learning-based methods for vastly simplifying the search problem specified in chemical design and drug discovery. In this work, I build upon this existing literature exploring the possibility of automatic chemical design and propose a novel generative model for producing a diverse set of valid new molecules. The proposed molecular graph variational autoencoder model achieves comparable performance across standard metrics to the state-of-the-art in this problem area and is capable of regularly generating valid molecule proposals similar but distinctly different from known sets of interesting molecules. While an interesting result in terms of addressing one of the core issues with machine learning-based approaches to automatic chemical design, further research in this direction should aim to optimize for more biochemically motivated objectives and be more informed by the ultimate utility of such models to the biochemical field.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Motivation

Chemical design and drug discovery are extremely difficult tasks for even the most experienced biochemists and chemical engineers. Estimates of the number of possible synthesizable molecules place the size of this discrete, unstructured search space on the order of the number of atoms of the universe or even greater (Schneider and Fechner, 2005). Actually synthesizing and rigorously testing any new molecule proposal is highly expensive and time-consuming, and discoveries of molecules with certain desirable properties do not necessarily inform us of which subsets of the entire space of synthesizable molecules would be best to explore next.

Within the last few years, some highly promising results from the intersection of the fields of machine learning and cheminformatics have demonstrated the possibility for learning an alternative search space in which discovery and optimization of molecule proposals is relatively trivial. By instead learning a continuous latent space representation of molecule space wherein useful properties are encoded on the different axes of variation, we may reframe the problems of chemical design and drug discovery as a much simpler search problem where promising proposals may be brought forth simply by taking advantage of local directional information, using more global information and prior examples of similar molecules, or by ideally making use of some combination of the two. In this manner, such learned latent representations of molecule space and the algorithms that enable learning the most informative and structured space representations possible hold great promise for automating the tasks of chemical design and drug discovery. Such automation could potentially revolutionize the manner in which and the ease with which biochemists and chemical engineers realize life-saving new drugs.

1.2 Contributions

In this work, I aim to build upon the rapidly growing body of work in automatic chemical design. My contributions may be briefly summarized as follows:

- I propose a novel neural network architecture for generating molecular graphs. In particular, I propose a novel variational autoencoder architecture designed to be particularly well-suited to learning a distribution over a latent representation of molecule space and to decoding valid molecule samples given samples from our latent distribution.
- I empirically demonstrate the strong performance of the proposed model on several standard benchmark molecule datasets, and place the performance of our model in context with several of the recently proposed approaches to this problem.
- I discuss some of the observed properties of the solutions found when incorporating each of the proposed model components into our architecture. In particular, I note the intriguing effects of incorporating modules for better accounting for variations in molecular complexity in our proposed latent variable deep generative model.
- I discuss the utility of the standard evaluation metrics for generative models of molecules, highlighting some concerns with the current implied goals in developing improved models for this problem area and proposing alternative directions for future work.

1.3 Organization

The remainder of this thesis is organized as follows. Chapter 2 provides a broad overview of the relevant background material for this work, discussing the general research areas of variational autoencoders and of graph neural networks. Chapter 3 discusses the most relevant previous work in automatic chemical design as each work relates to this thesis. Chapter 4 provides a more in-depth discussion of the proposed model, detailing the motivation behind and methodology underlying each of the model components incorporated in our architecture. Chapter 5 presents and discusses our experimental results on several standard benchmark molecule datasets for this task. Chapter 6 summarizes overarching takeaways from this work.

Chapter 2

Background

In this chapter, I discuss the theoretical results underpinning the proposed model and most pertinent for understanding the introduced approach to the problem of automatic chemical design. This work lies at the intersection of variational autoencoders (Kingma and Welling, 2013; Rezende et al., 2014) and graph neural networks (Gori et al., 2005; Scarselli et al., 2009), and thus the remainder of this chapter is organized around providing a high-level understanding of the derivations and recent developments for each of these general model frameworks.

2.1 Variational Autoencoders

2.1.1 Variational Inference

For any set of observed random variables $\mathbf{x}$, we might naturally express any relationships between these variables by means of determining the full joint distribution $p(\mathbf{x})$ between them. In its purest form, the fundamental task in probabilistic modeling involves inferring as much information about this joint distribution $p(\mathbf{x})$ as we possibly can. In practice, however, estimating $p(\mathbf{x})$ quickly becomes intractable; instead, it is often more effective to introduce the concept of latent, unobserved variables and to assume that the underlying dynamics governing $p(\mathbf{x})$ can be largely explained by a smaller set of latent variables $\mathbf{z}$. Given properly defined simplifying assumptions and a means by which to approximately integrate out $\mathbf{z}$, the latent variable model (LVM) approach to probabilistic modeling then provides us considerable power and flexibility to *infer* information about the underlying relationships between the observed variables $\mathbf{x}$.

To express this problem in more mathematical language, let us consider that any relationship between observed variables $\mathbf{x}$ and unobserved latent variables $\mathbf{z}$ may be described

by the broad concept of the joint distribution $p(\mathbf{z}, \mathbf{x})$ between them. By the product rule of probability, we know that we can decompose this joint as

$$p(\mathbf{z}, \mathbf{x}) = p(\mathbf{z}|\mathbf{x})p(\mathbf{x}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) \quad . \tag{2.1}$$

Rearranging terms, we note the following equivalence

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{z}, \mathbf{x})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x})} \quad . \tag{2.2}$$

We thus arrive at the fundamental theorem underlying Bayesian probability in the context of latent variable models. While a simple derivation from the basic rules of probability, Bayes' theorem allows us to isolate the distribution of interest, the posterior $p(\mathbf{z}|\mathbf{x})$ of our latent variables $\mathbf{z}$ given $\mathbf{x}$, and describe this distribution in terms of distributions we may *approximately* model and learn from data. In particular, in the framework of probabilistic modeling we are free to define the prior $p(\mathbf{z})$ of our defined latent variables $\mathbf{z}$ and to define an abstract conditional relationship of $\mathbf{x}$ on $\mathbf{z}$ by which we then learn from data our likelihood $p(\mathbf{x}|\mathbf{z})$. Given reasonable approximations of these distributions, we now have powerful tools through which to infer information about our posterior of the latent variables $\mathbf{z}$ given the observed variables $\mathbf{x}$ – or, in other words, through which to infer information about the underlying dynamics governing the behavior of our variables of interest.

In most interesting cases, the evidence or marginal likelihood term $p(\mathbf{x})$ in the denominator of (2.2) is intractable to estimate. We thus resort to methods for approximate posterior inference. In the school of Bayesian machine learning, there are two broad classes of approaches for this task: Monte Carlo methods, where we estimate complex probability distributions by taking a large number of unbiased samples such that our estimate is guaranteed to converge to the true posterior in the limit, and variational inference (VI) methods, where we approximate more complex distributions by maximizing some measure of "closeness" between our more interesting distribution and some defined, simpler distribution. For the purposes of this discussion, I focus on variational methods.

While Monte Carlo methods generally require a considerable amount of computation time and are prone to high variance estimates due to the large number of samples necessary to converge to an estimate of the posterior, variational methods frame posterior inference as an optimization problem wherein we aim to fit as tightly as possible some simple lower bound on our true posterior. To do so, variational approaches generally assume some family of parameterized distributions for which we optimize the distribution parameters to minimize some measure of similarity between our variational distribution and the true one. In this manner, variational methods trade off variance in posterior approximations for bias: unless the

true posterior is captured within the family of distributions we have assumed, our variational approximation will be biased in some form. Framing the problem in this manner also allows variational methods to arrive at posterior estimates with much lower computational cost than Monte Carlo methods.

To formally define the variational inference approach, we must first define a family of simple, parameterized distributions $q_\theta(\mathbf{z}|\mathbf{x})$, where θ represents the distribution parameters. Given this family of distributions, we may express the log marginal likelihood of our data as

$$\log p(\mathbf{x}) = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x})] \tag{2.3}$$

Note that this expectation can then be decomposed as

$$\begin{aligned}
\log p(\mathbf{x}) &= \mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{z}|\mathbf{x})}\right] \\
&= \mathbb{E}_{\mathbf{z}}\left[\log \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{z}|\mathbf{x})}\frac{q_\theta(\mathbf{z}|\mathbf{x})}{q_\theta(\mathbf{z}|\mathbf{x})}\right] \\
&= \mathbb{E}_{\mathbf{z}}[\log p(\mathbf{x}|\mathbf{z})] - \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\right] + \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})}\right] \\
&= \mathbb{E}_{\mathbf{z}}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) + D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x}))\,.
\end{aligned} \tag{2.4}$$

Rearranging terms, we finally have an expression for a lower bound on the log marginal likelihood:

$$\log p(\mathbf{x}) - D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x})) = \mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))\,. \tag{2.5}$$

The right-hand side of (2.5) is computationally tractable as long as we define the likelihood $p(\mathbf{x}|\mathbf{z})$, prior $p(\mathbf{z})$, and our variational posterior $q_\theta(\mathbf{z}|\mathbf{x})$ such that they can be tractably evaluated. While we cannot evaluate $D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x}))$ due to the presence of the unknown true posterior $p(\mathbf{z}|\mathbf{x})$, by the properties of the KL-divergence we know that this term must be nonnegative. By optimizing the expression on the right-hand side, then, we can optimize a lower bound on the evidence distribution $p(\mathbf{x})$. We thus have an expression for the evidence lower bound (ELBO) that we will aim to optimize in taking the variational inference approach to learning about the dynamics of our observed variables $\mathbf{x}$. Note that maximizing the ELBO amounts to an optimization balancing between two competing goals: assigning high likelihood to the observations given our latent variables and variational posterior, and encouraging a diffuse posterior distribution approximation close to our prior beliefs on the latent variables.

As an alternative derivation of the variational inference objective, let us come back to the inference problem defined by latent variable models. With LVMs, we are interested in inferring information about the true posterior $p(\mathbf{z}|\mathbf{x})$. From this perspective, variational inference can then be viewed as an optimization problem where we aim to determine the best approximation $q_{\theta^*}(\mathbf{z}|\mathbf{x})$ to $p(\mathbf{z}|\mathbf{x})$. Given some assumed family of simple parametric distributions $q_\theta(\mathbf{z}|\mathbf{x})$, we might naturally aim to minimize the KL-divergence $D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x}))$ from our true posterior to our variational approximation. However, directly optimizing this quantity is intractable. We thus instead note the following observation:

$$\begin{aligned}
D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x})) &= \mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\mathbf{x})}\left[\log \frac{q_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})}\right] \\
&= \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\theta(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{z},\mathbf{x})}\right] \\
&= \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z},\mathbf{x})} + \log p(\mathbf{x})\right] \\
&= \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z},\mathbf{x})}\right] + \mathbb{E}_{\mathbf{z}}[\log p(\mathbf{x})] \\
&= \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\theta(\mathbf{z}|\mathbf{x})}{p(\mathbf{z},\mathbf{x})}\right] + \log p(\mathbf{x}) \\
&= \mathbb{E}_{\mathbf{z}}[\log q_\theta(\mathbf{z}|\mathbf{x}) - \log p(\mathbf{z},\mathbf{x})] + \log p(\mathbf{x}) \\
&= \mathbb{E}_{\mathbf{z}}[\log q_\theta(\mathbf{z}|\mathbf{x}) - \log p(\mathbf{x}|\mathbf{z}) - \log p(\mathbf{z})] + \log p(\mathbf{x}) \; . \qquad (2.6)
\end{aligned}$$

The log marginal likelihood of the observed variables $\log p(\mathbf{x})$ is constant with respect to variational parameters θ, thus minimizing $D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x}))$ with respect to θ is equivalent to minimizing the expectation $\mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\mathbf{x})}[\log q_\theta(\mathbf{z}|\mathbf{x}) - \log p(\mathbf{x}|\mathbf{z}) - \log p(\mathbf{z})]$ with respect to θ. Minimizing this quantity is equivalent to maximizing its negation, thus giving us the standard expression for the variational lower bound

$$\begin{aligned}
&\min_\theta \; \mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\mathbf{x})}[\log q_\theta(\mathbf{z}|\mathbf{x}) - \log p(\mathbf{x}|\mathbf{z}) - \log p(\mathbf{z})] \\
&\qquad \equiv \max_\theta \; \mathbb{E}_{\mathbf{z}}[\log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}) - \log q_\theta(\mathbf{z}|\mathbf{x})] \\
&\qquad \equiv \max_\theta \; \mathbb{E}_{\mathbf{z}}\left[\log p(\mathbf{x}|\mathbf{z}) + \log \frac{p(\mathbf{z})}{q_\theta(\mathbf{z}|\mathbf{x})}\right] \\
&\qquad \equiv \max_\theta \; \mathbb{E}_{\mathbf{z}}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \; . \qquad (2.7)
\end{aligned}$$

Thus have we once again derived our original expression for the variational or evidence lower bound. Variational inference may then be viewed as a class of methods for performing

approximate inference in the context of highly complex distributions wherein, given some assumed family of much simpler distributions, we aim to directly optimize some lower bound on the true data distribution $p(\mathbf{x})$ or we aim to obtain an optimal approximation to the true posterior of our defined latent variables $\mathbf{z}$ given the observed variables $\mathbf{x}$. In this manner, VI defines approximate inference as a (biased) optimization problem requiring considerable less computation than Monte Carlo methods and suffering from considerably less variance in estimates produced. As we shall see for the example of variational autoencoders, framing approximate inference as an optimization problem makes the VI approach particularly suitable to combination with the high capacity of neural networks for function approximation.

2.1.2 Deep Autoencoders

Much of the recent success of neural networks has been driven by the remarkable performance of different neural network architectures on difficult supervised learning problems. Applications to traditional unsupervised learning problems, however, have been relatively limited. One notable architecture for learning on unlabeled data is that of the deep autoencoder. The autoencoder framework defines a general architecture that is trained to reconstruct the input as "closely" as possible. In order to prevent the model from learning the undesirable trivial solution of the identity mapping from input to output, we constrain the autoencoder to reconstruct the input given (most commonly) either an "information bottleneck" layer or a corruptive noise factor on the inputs. For the purposes of this discussion, I shall focus on the information bottlenecking variant of deep autoencoders.

Intuitively, bottlenecking autoencoders are based on the idea that the simplest explanation of the observed variation in the data is likely the best. A bottlenecking or sparse autoencoder forces the input data to be passed through a layer of lower (effective) dimensionality than the original input, thus in this manner learning a lower-dimensional "latent manifold" representation of the data onto which the neural network "encodes" input samples and from which samples in the original data space may be reconstructed. In this manner a bottlenecking autoencoder may be thought of as a compression or information extraction technique, though generally with a much more complex, nonlinear basis than more traditional techniques. It should be noted that this model specification does not define a proper generative model: while new samples can be encoded to and decoded from the learned latent manifold, reconstructions will be forced to be quite similar to the data observed during training.

2.1.3 Variational Autoencoders

From the perspective of probabilistic modeling, variational autoencoders (Kingma and Welling, 2013; Rezende et al., 2014) might simply be considered a particularly flexible and scalable variational technique for approximate posterior inference. Mathematically, variational autoencoders (VAEs) may be easily formulated directly from the derivation of the variational inference optimization objective. From the perspective of deep autoencoding models for unsupervised learning problems, it might be more interesting to observe that VAEs generalize the standard bottlenecking autoencoder architecture so that, instead of defining a rather simple (but powerful) compression technique, we have defined a proper generative model from which we might sample interesting new data points. While standard autoencoders assume data points lie on some lower-dimensional latent manifold and thus force data points to lie on this manifold, VAEs assume that data points are generated by sampling from an underlying, lower-dimensional latent distribution, thus defining a relatively relaxed assumption on the dynamics of the underlying latent space and thus allowing us to observe new data samples reasonably differentiated from the original training data.

To discuss both perspectives on the variational autoencoder framework, let us begin by returning to the definition of the optimization objective in variational inference. Recall that, given a definition of some latent variables $\mathbf{z}$ and of a tractable family of parameterized distributions $q_\theta(\mathbf{z}|\mathbf{x})$ over which to optimize, in variational inference we aim to maximize the evidence lower bound (ELBO) defined as

$$\mathscr{L}_{ELBO} = \mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \tag{2.8}$$

given some likelihood $p(\mathbf{x}|\mathbf{z})$ and prior $p(\mathbf{z})$ definitions that we may properly evaluate. Note that, should we define the likelihood $p(\mathbf{x}|\mathbf{z})$ to be a learnable parameterized function, then this optimization objective naturally lends to being formulated in terms of a deep autoencoder network: if we let θ represent the joint set of parameters of some encoding neural network and of the assumed variational distribution and we let ϕ represent the parameters of a corresponding decoding network, then jointly learning a variational posterior $q_\theta(\mathbf{z}|\mathbf{x})$ with a parameterized likelihood distribution $p_\phi(\mathbf{x}|\mathbf{z})$ might simply be framed as learning a deterministic function mapping input samples into parameter estimates of our latent distribution jointly with a second deterministic function mapping samples taken from this latent distribution back to the original input data samples. In this way we may frame the variational inference objective as a task wherein we aim to accurately reconstruct the training data from a set of lower-dimensional latent representations of the original inputs

while also accounting for some regularization factor encouraging the encoding-decoding function learned to be more capable of generalizing to new data.

One large issue with this formulation remains. In projecting input samples into a latent distribution rather than onto a latent manifold, variational autoencoders define a latent layer that is now stochastic and from which we must sample for decoding. Backpropagating through a naive definition of such a layer is poorly defined. To address this issue, let us consider a formulation of the VAE framework where we assume that the variational posterior $q_\theta(\mathbf{z}|\mathbf{x})$ is a multivariate Gaussian distribution with a diagonal covariance matrix, and the prior $p(\mathbf{z})$ is the standard zero-mean multivariate Gaussian distribution with an identity covariance matrix. Sampling from $q_\theta(\mathbf{z}|\mathbf{x})$ may then be performed as

$$q_\theta(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \mu(\mathbf{x}), \sigma^2(\mathbf{x})\mathbf{I}) = \mu(\mathbf{x}) + \sigma^2(\mathbf{x}) \odot \varepsilon \; ; \qquad \varepsilon \sim p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.9}$$

where latent distribution parameters $\mu(\mathbf{x})$, $\sigma^2(\mathbf{x})$ are outputted by the encoder network given the input data $\mathbf{x}$. By applying this "reparameterization trick," given a set of input samples $\mathbf{x}$ and a draw of samples ε from the prior $p(\mathbf{z})$ we can then express the mappings from $\mathbf{x}$ to $\mathbf{z}$ and from $\mathbf{z}$ back to $\mathbf{x}$ as deterministic, differentiable functions. We have thus specified a model enabling stable end-to-end learning of a variational approximation to the true posterior jointly with a reconstructive likelihood distribution that enables sampling of new data samples simply by sampling from $p(\mathbf{z})$. Note further that, with the particular specification of diagonal Gaussian distributions for $q_\theta(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{z})$, calculating the KL-divergence $D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$ can be performed in closed form.

Variational autoencoders thus define a latent variable model where we make use of the stable deep bottlenecking autoencoder architecture to jointly learn a variational approximation $q_\theta(\mathbf{z}|\mathbf{x})$ to the true posterior $p(\mathbf{z}|\mathbf{x})$ and a reconstructive likelihood $p_\phi(\mathbf{x}|\mathbf{z})$ of the observed data given our current estimates of the latent variables. It should also be noted that, by assuming that observed variables $\mathbf{x}$ are generated by samples from some distribution over latent variables $\mathbf{z}$, VAEs define a latent layer where, due to the noise in the samples generated from the latent layer, data points distant from each other in the original data space cannot be properly reconstructed by generating very "close" latent samples to be passed to the decoder. The latent mapping, then, is encouraged to be relatively diffuse so that interpolation in the latent space is relatively interpretable and produces variation between samples relatively well correlated with distance in the latent space. The diffuseness of the latent posterior can also be observed by directly considering the terms in the optimization objective (2.8): $\mathbb{E}_{\mathbf{z}\sim q_\theta(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})]$ can be thought of as a "reconstruction term" of the data given the latent samples where optimizing this term corresponds to encouraging the model to assign high probability to latent samples that best reconstruct the original training data,

while $D_{KL}(q_\theta(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$ serves as a form of a regularization term encouraging the latent distribution to spread probability mass over the latent space.

2.2 Graph Neural Networks

Perhaps one of the most interesting data structures on which we might aim to perform learning and inference tasks is that of the general graph. General graphs are natural abstract representations of essentially any interacting system with multiple factors, agents, or any sort of reasonably individualized unit. Graph representations enable the organization of relational information at roughly any level of abstraction that might be best suited for a particular reasoning task. Thus, developing algorithms that enable learning on general graphs is a highly interesting research direction both for the potential implications on modeling capabilities across a wide variety of problem domains and for the general value of enabling a better understanding of intelligence and reasoning at higher levels of abstraction.

Research on neural network models for general graph structures largely developed along two distinct tracks: the spatial approach and the spectral approach. In this subsection, we discuss the primary technical details and modeling assumptions taken by both approaches, and further discuss many of the recent developments in this quickly growing research area.

2.2.1 Spatial Methods

In the seminal work for this research problem, Gori et al. (2005) and Scarselli et al. (2009) proposed the original spatial approach to defining a functional "graph neural network" (GNN) model. Given the simple observation that the nodes in a graph can be viewed as abstract representations of any particular objects or concepts of interest and the edges between these nodes representations of the relationships between them, these works proposed a general learning algorithm where each node in a graph is associated with a vector representation that is successively learned by applying a parametric transition function dependent on a given node and its neighbors for several iterations until convergence. For this particular model, to guarantee convergence it is important that the parameterized transition function amounts to a contraction mapping between successive iterations. Learning and prediction are then possible by applying a parameterized output function to the converged node representations. In this manner, the spatial approach to graph neural networks may be thought of as an extension of recurrent neural networks from directed chain graphs to any arbitrary graph type. In the spatial definition of GNNs, we make use of the relationships defined by the graph structure for performing several steps of "information propagation" in order to update

state representations of each node until converging towards a representation deemed suitable enough for the ultimate prediction task.

More recently, Li et al. (2015) proposed a gated graph sequence neural network (GGSNN) model building directly off of the spatial approach first proposed by Gori et al. (2005). This work takes inspiration directly from the observation that Gori et al.'s original GNN model could be seen as a generalization of the more common recurrent neural network architecture. In this case, Li et al. built off of the gated recurrent unit (Cho et al., 2014) version of RNNs, proposing a simple gated graph neural network (GGNN) module with which propagation is now performed across a given graph by a fixed number of rollout steps of the recurrent unit to produce the final output prediction. Several of these GGNN modules may then by combined in a sequence to produce a similar output model to that of Gori et al. As a further algorithmic improvement, it should be noted that the GGSNN architecture removes the constraint placed upon the original GNN architecture that function operations between rollout steps be contraction mappings.

2.2.2 Spectral Methods

Much of the recent success of deep learning could be largely attributed to the success of convolutional neural networks (CNNs) on a variety of difficult image-based, video-based, and even language-based tasks. While one could argue that much of the recent developments in deep learning amounts to the rediscovery of algorithms originally derived decades prior, one would be hard-pressed to counter the assertion that much of the recent work developing ever more complex and clever convolutional architectures for difficult vision tasks has helped lead to a fundamental paradigm shift in both computer vision and machine learning research. In many other subareas of machine learning research, perhaps there may be insights to be gleaned by taking inspiration from the techniques that led to the success of CNNs and adapting them for an altogether different problem domain.

This exact idea – generalizing the most successful techniques used in convolutional architectures – is the underlying motivation for the spectral approach to developing graph neural networks. Images, videos, and even written text can all be viewed in some sense as data formed as highly regular graph structures. Essentially, each could be viewed as a grid-like structure in one, two, or three dimensions. If convolutional and pooling operations can be efficiently and properly defined on this subset of graph structures, one interesting research direction might be to generalize such operations to any arbitrary graph.

This view, however, ignores one of the largest fundamental issues with such a generalization: convolutional architectures work well in the domain of computer vision tasks where the data space is extremely high-dimensional by taking advantage of many of the inherent

regularities of image-based data – regularities that would not hold for general graph data. In particular, natural image data inherently lies within a Euclidean space where directions and distances between points have meaning and are well-defined. General graphs, on the other hand, lie in a space with no such notions defining the relationships between different graph data points. Furthermore, convolution and pooling work well on image-based tasks by taking advantage of three fundamental properties of natural images: locality of relevant information for a particular feature, stationarity of concepts or invariance to translation of object classes across an image, and a natural hierarchy of informative feature representations for learning more and more complex features in a compositional manner. Convolution and pooling on natural images can thus be applied highly efficiently and effectively, enabling ever deeper architectures conducive to more difficult image-based tasks.

In defining similar operations for general graphs, we need (1) to determine an approach to general graph structures that allows us to maintain these assumptions of locality, stationarity, and hierarchically structured features and (2) to define fast convolution and pooling operations for general graphs. As Bruna et al. (2013) first put forth, spectral graph theory provides a manner by which to generalize convolutions to graphs. While a full explanation of the mathematics underlying the spectral approach is beyond the scope of this work, the approach that Bruna et al. defined is fundamentally based on an eigendecomposition of the graph Laplacian as a means of determining a smooth orthonormal basis for graphs. Convolutional filters are then determined by taking the first n eigenvectors of the graph Laplacian, thus allowing for capturing the smooth geometry of the graph. Smoothness in the spectral domain corresponds to locality in the spatial domain, thus allowing us to preserve this property for graphs. As originally formulated, spectral graph convolution networks suffered from a number of substantial issues: the parameters per layer grow linearly with the input data size, calculation of the eigendecomposition is highly computationally expensive, and filter coefficients are dependent on the basis and thus different for each graph. Defferrard et al. (2016) then improved upon the spectral approach derived by Bruna et al. (2013) by proposing a graph convolution operation based on orthogonal Chebyshev polynomials and by proposing a fast graph coarsening method as a generalization of pooling to graph structures. Defferrard et al. proposed to represent smooth spectral filters in terms of Chebyshev polynomials of the Laplacian eigenvalues, thus producing a relatively stable orthonormal basis while circumventing the need to compute the eigendecomposition of the graph Laplacian. Defferrard et al. further observed that downsampling on graphs would be equivalent to graph coarsening or graph partitioning, problems for which there already existed an extensive literature from graph theory. In general, graph partitioning is an NP-hard problem, thus Defferrard et al. proposed a fast approximate method based on finding balanced

cuts and heavy edge matching where a balanced binary search tree is built out for the multiple levels of graph coarsening and vertex indices are rearranged so that adjacent indices are pooled together at subsequent levels. In this manner, the authors propose a graph pooling mechanism as efficient as standard grid pooling for one-dimensional Euclidean data.

Kipf and Welling (2016) then proposed the standard modern variant of graph convolutional networks (GCNs) as a faster approximation to the method proposed by Defferrard et al. (2016). By taking a simple first-order approximation to the Chebyshev polynomial basis proposed by Defferrard et al. and incorporating some additional renormalization techniques for stable learning, Kipf and Welling proposed the first graph convolutional architecture able to achieve state-of-the-art performance on difficult real-world learning and prediction tasks. It is this formulation of the graph convolutional network that I apply in deriving an efficient architecture for our model and generative task.

2.2.3 Modern Approaches

"Modern" approaches to developing graph neural networks could likely be traced back to Gilmer et al. (2017). In this work, Gilmer et al. unified the previous research directions explored in developing GNNs and demonstrated that many of the proposed models amounted to different formulations of the same more abstract framework of "neural message passing." In the general formulation of message passing neural networks (MPNNs), learning can be described in terms of two distinct phases: a message passing phase and a readout phase. For the case where we are working with undirected graphs $G = (V, E)$ where nodes $v \in V$ are represented with feature vectors x_v and edges are similarly provided attached vectorized feature representations e_{vw}, the message passing phase may be described in terms of

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_W^t, e_{vw}) \tag{2.10}$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{2.11}$$

where message passing is performed by iteratively applying these functions for T time steps, M_t defines some arbitrary function controlling message passing and U_t similarly defines a function controlling the update of node feature representations, h_v^{t+1} corresponds to the intermediate node representations at each update step and m_v^{t+1} corresponds to the messages passed for determining updated node representations, and $N(v)$ denotes the neighborhood of node v. Given the final node representations h_v^T, the readout phase is then described by

$$\hat{y} = R(\{h_v^T | v \in G\}) \tag{2.12}$$

where $\hat{y}$ represents some output prediction and R defines a function dependent on all the nodes in G for producing the output prediction. Previous work on both the spatial and spectral approaches could then be described in this framework with different definitions for M_t, U_t, and R. In this manner, Gilmer et al. demonstrated that deep learning on graphs ultimately amounts to some form of approximate message passing to calculate useful updated feature representations for whichever inference or prediction task might be of interest given a particular dataset of graphs. Thus, rather than deriving further theoretical justification for a particular approach to MPNNs, perhaps it might be better to focus on deriving new variations as proves particularly suitable for some relevant problem domain.

Chapter 3

Related Work

3.1 Prior work in automatic chemical design

Within the field of machine learning, the task of automatic molecule and chemical design is a relatively nascent but rapidly growing area of research. In this section, I briefly review some of the most relevant previous work in this problem domain.

3.1.1 SMILES-based Approaches

In the seminal work on automatic chemical design and drug discovery, Gómez-Bombarelli et al. (2016) proposed framing the task as an optimization problem wherein we might circumvent the combinatorial issues arising from optimizing directly in molecule space by first learning a continuous, structured latent space representation of molecule space. In approaching the task in this manner, Gómez-Bombarelli et al. introduced a modified recurrent variational autoencoder model operating on SMILES-based character sequence representations of molecules (Weininger, 1988). After first passing the SMILES encodings through an RNN-based encoder to output continuous latent vector representations, the proposed "ChemVAE" model then passed latent samples to both a property predictor module and a recurrent decoder module so as to learn a latent representation wherein optimizing in certain directions might allow for optimizing new molecule samples with respect to a certain desirable property.

While a promising and thought-provoking first result providing an entirely new framework for approaching drug discovery, this work – and all subsequent work taking a SMILES-based approach – suffered from a number of issues. SMILES codes were originally designed as convenient, compact representations of molecules for more traditional cheminformatics tasks and approaches. When used directly as the molecule representation passed into a variational

autoencoder, however, SMILES codes prove decidedly suboptimal for the learning task involved. The SMILES coding system was not originally designed so that similar code representations necessarily correspond to similar molecules; as such, a small change in the SMILES string representation of a molecule could correspond to a drastic change in the chemical viability and underlying properties of the resultant molecule. Moreover, one of the primary issues observed by Gómez-Bombarelli et al. with their approach was the rather low likelihood of generating valid new molecule proposals when sampling new SMILES code sequences. The authors thus conjectured that – should it ultimately prove possible to design efficient learning algorithms for such structures – graph-based representations of natural molecules would make for a much more expressive representation form. Such representations would likely better allow for future ChemVAE-like approaches to learn essential properties of molecules like basic chemical validity of new proposals.

Despite the many limitations of the underlying representation form, several notable results taking a SMILES-based approach followed from Gómez-Bombarelli et al.'s seminal work. Research in this direction has largely amounted to applying more sophisticated techniques and more complex modeling approaches than the basic, general ChemVAE framework. In particular, by proposing a highly general generative model, Gómez-Bombarelli et al. aimed to develop a model that might learn from data to encode the necessary rules in molecule reconstruction. In experiments, however, the proposed model had learned a continuous latent space with several "dead regions" from which molecule samples largely proved invalid – either as valid graphs in the more mathematical sense or as chemically plausible graphs in consideration of the particular problem domain. A natural direction for further improvement then would be to design a more constrained generative model directly incorporating existing knowledge relevant to the problem domain.

"GrammarVAE" (Kusner et al., 2017) then followed directly from ChemVAE, proposing to address the "dead region" issue and the inherent fragility of SMILES-based representations by incorporating context-free grammars (CFGs) into the VAE architecture. By incorporating a grammar into the generative encoding-decoding process, the proposed model was designed to make easier the involved learning task by explicitly removing the necessity of learning the syntactical rules involved in converting molecule graphs to proper SMILES codes. GrammarVAE defined an autoencoding probabilistic model that operated on "production rules" extracted from parse trees of a particular character sequence as defined by a given grammar. In experiments, the proposed GrammarVAE model demonstrated significant improvements in both generating valid new molecule samples and in learning a more structured latent space. However, using CFGs for constraining the molecule generation process proved to be too restrictive: while new generated samples represented valid SMILES codes, the usage of

CFGs did not directly allow for learning a model that encoded the *semantics* of the underlying language – thus often resulting in molecule proposals that were unrealistic according to known rules of chemical bonding.

"SD-VAE" (Dai et al., 2018) thus subsequently defined a VAE for sequence data able to generate new sequence outputs that are both syntactically correct and semantically coherent. The proposed "syntax-directed VAE" model encodes and decodes sequence data as guided by both a context-free grammar and an "attribute grammar " capturing semantic information. Semantically coherent molecules are then generated by building out an attribute tree of stochastically selected and inherited attributes as constraints for generating syntactically correct molecule production rules. In experiments, SD-VAE demonstrated a considerably stronger ability to optimize generated samples for desirable molecular properties and to produce a diverse set of new samples. However, the performance of this modeling approach significantly relies on the experimenter's ability to define a proper constrained search space by means of both optimal syntactical and semantic grammar rules, where for many interesting problems the latter requirement can be quite difficult to fulfill properly.

Rather than take the more constrictive approach of learning a proper latent distribution from which to generate new molecule samples, "ORGAN" (Guimaraes et al., 2017) defined an entirely different probabilistic modeling approach than ChemVAE: learn an implicit distribution instead and take advantage of algorithms better designed for producing natural-looking samples. In this work, Guimaraes et al. proposed training a Wasserstein GAN (Arjovsky et al., 2017; Gulrajani et al., 2017) for discrete sequences by taking advantage of the REINFORCE algorithm (Williams, 1987, 1992) for performing gradient estimation of non-differentiable functions. In this manner, ORGAN was designed to be a generative model particularly well-suited to working with discrete sequence representations and differentiating through discrete sampling steps. By then further incorporating auxiliary reinforcement learning-based objectives for optimizing molecule proposals for particular desirable properties, the proposed "objective-reinforced GAN" model also proved particularly well-suited to generating realistic molecule samples optimized for certain pre-defined objectives. Nevertheless, while ORGAN cleanly side steps some of the issues with a VAE-based approach to this problem, learning an implicit distribution of molecules ultimately leads to forfeiting much of the valuable information provided in learning a valid latent distribution of molecule space and particularly makes it complicated to optimize newly generated samples after training.

"SSVAE" (Kang and Cho, 2018) also recently defined an interesting semi-supervised extension to Gómez-Bombarelli et al.'s original ChemVAE model that more concretely incorporates molecule properties in the decoding process for improved conditional generation. In particular, the "semi-supervised VAE" architecture defines a model where molecule

properties are treated as an additional latent variable upon which to directly condition the generation of both latent samples and reconstructed molecules. Property prediction is based directly on the original input molecule. Thus, in the context where we have access to a dataset of molecules labeled with certain desirable chemical properties, Kang and Cho's proposed SSVAE model provides a useful formulation for better generating new molecule samples with desirable properties.

3.1.2 Graph-based Approaches

In perhaps the original work taking a graph-based approach to automatic chemical design, Simonovsky and Komodakis (2018) proposed a VAE model directly operating on entire graph structures, as represented by an adjacency matrix and vector representations of nodes and edges. In particular, the authors proposed to decode latent samples in a single generation step outputting the full graph. In experiments, Simonovsky and Komodakis' "GraphVAE" model validated Gómez-Bombarelli's original conjecture: using direct graph representations of molecules rather than SMILES codes drastically improved the model's ability to generate valid molecules. While a promising first result in this direction, due to the single-step generation process used the GraphVAE model ultimately proved too computationally expensive and could only be applied to the smallest molecular graph structures.

Subsequent work has largely focused on defining more scalable molecule graph generation processes. The two most relevant results to this work proposed novel probabilistic iterative graph generation models largely aiming to tackle this exact issue. Li et al. (2018a) proposed a particularly simple and general generative model of graphs where new graph proposals are generated sequentially in an iterative stochastic process governed by parameterized distributions determining whether to add a node, add an edge, or terminate the generation process at each step. In their proposed architecture, Yujia Li et al. make use of the gated graph neural network module (Li et al., 2015) for parameterizing the distributions and handling the changing size of the input graph at each step of the generation process, and train the model to learn an overarching distribution of graph structures by sampling multiple orderings of graph generation for each graph proposal. Li et al. (2018b), on the other hand, proposed a similar iterative generative model of graphs specifically designed for the biochemical domain. In this work, Yibo Li et al. proposed a more constrained generation process where at each step the model could propose bonding a new atom to the existing graph, bonding two existing atoms together, or terminating the generation process. The authors also proposed using graph convolutional networks (Kipf and Welling, 2016) as a more computationally efficient and scalable computation module in their architecture.

While both approaches demonstrated strong results in terms of the quality, number, and novelty of valid molecule samples generated, the iterative generation processes defined introduce new modeling issues that must be accounted for. Both Yujia Li et al. and Yibo Li et al.'s proposed models require a substantial number of expensive generation steps for proposing new molecule samples, and both ultimately could scale better with increasing complexity of molecule graphs. Perhaps most importantly, these types of iterative generation processes are not ideally suited for generating naturally isomorphic structures. In estimating the distribution over graphs in particular, such iterative procedures require an expensive process of approximately integrating out some defined distribution over decoding routes. Each graph generation process could also be further improved by conditioning on samples from a latent representation of molecule space.

Several more recent approaches proposing more scalable latent variable deep generative models of graphs bear noting as well. "NeVAE" (Samanta et al., 2018) defined a variational autoencoder model of molecular graphs designed particularly for addressing issues related to graph isomorphism and the variability in graph sizes. In this work, Samanta et al. proposed to learn a latent space representation of nodes or atoms instead of entire molecules, and thus move much of the model complexity into the decoder. Decoding in this model is then performed by first sampling a number of nodes and edges from parameterized Poisson distributions and sampling edge connections from a single parameterized multinomial distribution. Liu et al. (2018) then followed up with a constrained graph VAE (CGVAE) model involving a similar decoding process and incorporating ideas from Li et al. (2018a)'s proposed model. In the CGVAE architecture, decoding is performed by first initializing a set of possible nodes to connect in forming the final graph sample, given some fixed max number of nodes to generate in a graph sample. Given this set of unconnected nodes, the decoder then iterates over the given nodes, performs a step of edge generation and associated edge labeling for the currently selected node, passes the current connected molecule graph to a GGNN module for propagation and updating the node representations, and repeats this process until a special termination node has been selected in the edge generation step. This entire process is then repeated for a new node in the current connected graph unless there are no valid candidates in the connected graph. As part of this process, to help ensure valid molecule generation the decoder also makes use of a "valency mask" to prevent generation of additional bonds on atoms that have already been assigned the maximum number of bonds for that particular atom type that one might naturally observe in a stable molecule.

"MolGAN" (De Cao and Kipf, 2018) also recently defined a deep generative model of graphs essentially adapting Guimaraes et al.'s proposed ORGAN model for graph structures instead of SMILES codes. The MolGAN framework proposes learning an implicit

distribution of molecular graphs while training with auxiliary reinforcement learning-based objectives, most notably one corresponding to a reward function returning positive signals for generating valid molecule proposals. In the context of learning generative models of graph structures, MolGAN circumvents issues with previous likelihood-based methods arising from either the isomorphism of graph structures when applying sequential generation methods or computational constraints from one-step generation approaches. While this approach demonstrated particularly strong results in generating valid molecule samples, it faces similar limitations to that of the previous ORGAN model in choosing to learn an implicit distribution rather than an explicit one. Additionally, experiments demonstrated that this particular model formulation also suffered considerably from the well-known "mode collapse" issue with generative adversarial networks.

One other recent result of note in this direction is that of Jin et al. (2018)'s proposed "junction tree VAE" (JT-VAE) model. In this work, Jin et al. asserted that generating molecule sequences atom-by-atom is fundamentally a poor modeling approach: while character-level sampling might work well in language generation contexts, in the context of molecular chemistry this approach forces the model to generate several invalid intermediate molecules to hopefully arrive at a valid final sample. As such, JT-VAE defines an alternative, more chemically valid generation process wherein molecules are represented both as graph structures and as a "junction tree" of bonding steps of molecular subgraphs. Decoding then becomes a process of traversing a junction tree sample as a guiding mechanism for producing the ultimate molecular graph sample. In experiments, the authors achieve perfect validity results in newly generated samples while also demonstrating a strong ability to discover new molecule samples optimized for some property score. The JT-VAE framework, however, requires rather significant overhead: valid steps in building out the junction trees used in this model must be extracted from some training set, and the performance of the decoding significantly depends on the quality of these extracted steps.

Chapter 4

Methodology

In this chapter, I discuss the proposed model. Building off of much of the recent literature in deep generative modeling, graph neural networks, and machine learning-based approaches to the general problem of chemical and molecule design, I propose a variant of a variational autoencoder model particularly well-suited for molecular graph structures.

4.1 Proposed Model

In this work, the primary aim was to expand upon the nascent machine learning research area of graph-based approaches to automatic chemical design. While the literature on SMILES character sequence-based approaches extends back multiple years, only recently have graph-based approaches begun to appear. As posited in the original SMILES-based work on this problem (Gómez-Bombarelli et al., 2016), graph structures make for a much more natural representation of molecules and are not prone to many of the syntactical and semantic issues that arise with representing molecules with relatively fragile SMILES codes. A proper latent variable deep generative model of graphs would thus likely make for a strong modeling approach to the problem of generating and optimizing new molecule proposals as originally framed in Gómez-Bombarelli et al. (2016).

The proposed model directly builds off of the sequential graph generative model proposed by Li et al. (2018b). In particular, the overarching proposed framework essentially extends the generative process proposed by Li et al. (2018b) so that generation steps are conditioned on latent samples generated by stochastically sampling from a latent distribution parameterized by an encoder architecture taking as input molecular graph samples. In this manner, I propose a variational autoencoder model of graphs where decoding of latent samples is performed by means of the sequential generative process proposed by Li et al. (2018b).

4.1.1 Decoding Process

In Li et al. (2018b), the authors propose an iterative graph generation process particularly well-suited to generating molecular graphs. In comparison to similar graph generative models, Li et al. (2018b) significantly improves over the approaches proposed by Simonovsky and Komodakis (2018) and Li et al. (2018a) in terms of scalability and the size of the space of molecules the model is able to process. Li et al. (2018a) restricted experiments with their proposed model to molecules from the ChEMBL dataset of at most 20 heavy atoms; Simonovsky and Komodakis (2018) restricted their experiments to the substantially smaller and less complex QM9 and ZINC datasets. Yibo Li et al.'s proposed model, on the other hand, demonstrated strong results on a dataset of ChEMBL molecules with as many as 50 heavy atoms. The proposed model thus builds off of Yibo Li et al.'s sequential molecular graph generative process.

Fig. 4.1 details the overarching generative process proposed by Yibo Li et al. In their construction, molecules are represented as graphs $G = (V, E)$ where the vertex set V represents atoms in a molecule and the edge set E bonds between atoms. The set of possible vertices and possible edges are restricted to the chemical domain. Vertices are represented as one of a particular set of atom types, where atom types are defined as tuples of three numbers representing a particular atom's atomic number, number of explicit hydrogens attached to that atom, and number of formal charges associated with the atom. Edges then are restricted to be one of four bond types: single, double, triple, or aromatic. Let A and B refer to the sets of atom types and bond types, respectively, considered in this model.

Given these definitions of molecule representation and of the space of possible generation steps, Yibo Li et al.'s proposed generation procedure proceeds as follows. Starting initially from an empty graph, at the first step in a discrete sequence of time steps a molecule proposal is initialized to a single atom, where the choice of the atom type is sampled from a probability distribution parameterized by a neural network. At all subsequent time steps, either append, connect, or termination steps are sampled from another distribution parameterized by another neural network. In this context, append steps are defined as selecting a particular atom in the existing graph to which to attach a chosen type of atom-bond pair; connect steps are defined as adding a chosen bond type between two existing atoms in the current graph; and termination steps are simply a signal to end the generation process and return the final molecule sample. As illustrated in Figure 4.1, append steps are sampled from a tensor output probability predictions of dimensions defined by the number of possible atom types $|A|$, number of possible bond types $|B|$, and number of vertices in the current graph $|V_i|$. Connect steps are sampled from a separate matrix output of probability predictions of dimensions defined by the number of bond types and number of vertices in the current graph.

The model is trained to maximize the log-likelihood of graph samples generated. In order to handle issues arising from modeling a naturally isomorphic structure as a sequence of generation steps, in calculating the log-likelihood of a batch of samples the authors propose sampling a number of generation sequences for a given graph in order to marginalize out the distribution over sequential representations for a given graph.

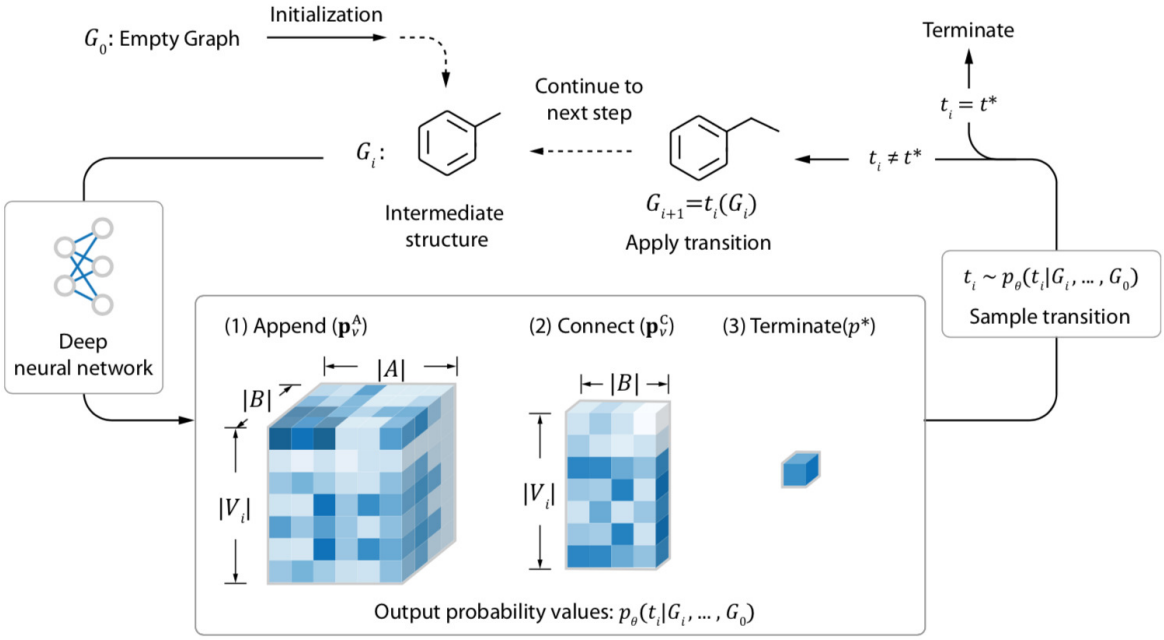


Fig. 4.1 Iterative molecule generation process used in our decoder. *Figure Credit:* Li et al. (2018b)

4.1.2 Deep Recurrent Attentive Writer (DRAW)

In Gregor et al. (2015), the authors proposed a more complex variant of the basic VAE model that was designed to be particularly well-suited to generating more complex and realistic image samples than possible with previous VAE formulations. As intuition, the proposed DRAW model is inspired by the notion that the generation of entire visual scenes in a single step is likely a more difficult learning task than is really necessary. A more natural approach might be to approximate the human creative process: allow for the generation of images in a sequence of gradual refinement steps, where at each step we focus on a particular subpatch of the canvas to further refine. In this manner, we define a generative process wherein the sequence of previous generation steps can be used to help decide what might most naturally be added to the scene next. Applied to the natural language context, such a model might be akin to the general editing process wherein an essay goes through several iterations of rough drafts and edits before arriving at the final product.

Fig. 4.2 shows a comparison of the basic variational autoencoder architecture and the more complex DRAW model. Making use of recurrent architectures for the encoder and decoder in lieu of the more standard feedforward modules, the DRAW architecture first initializes a "blank canvas" onto which to draw the reconstructed input. At each step, the architecture "reads" in a subpatch of the original image as the aspect of the scene to focus on next for generation, where the selection of the next read patch is conditioned on the current state of the output canvas and on the sequence of previous generation steps. "Writing" out new information to the output canvas is then dependent on the currently selected read patch.

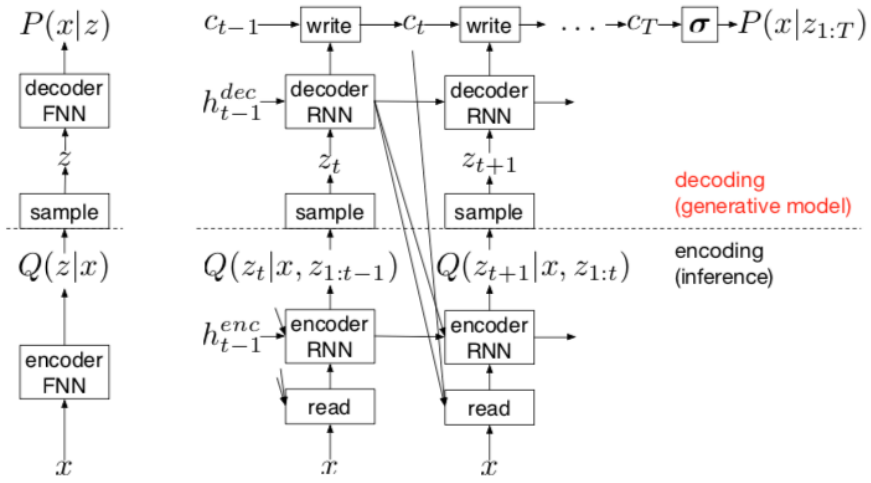


Fig. 4.2 Comparison of the vanilla VAE and DRAW architectures. *Figure Credit:* Gregor et al. (2015)

In the proposed model, I primarily focus on adapting the attentive reading and writing modules proposed in the DRAW model for an iterative generative model of molecular graphs. The size of natural and interesting molecules can differ considerably in terms of the number of atoms and bonds in different molecules. For a model where generation of molecular graph proposals is performed in an iterative manner, encoding and sampling latent representations of differently sized input molecules in a single step likely makes for a suboptimal approach to learning a valuable latent distribution of molecules. Furthermore, in conditioning the decoding process on samples taken from this latent distribution, conditioning would likely prove more informative by designing a manner of writing out the parameters of the latent distribution so that properties related to molecular complexity are better encoded. Similarly, it would likely also be beneficial to define a manner of reading from latent samples so that in each decoding step the model can focus in on the most pertinent latent parameters for determining which append or connect step would be best applied next.

Defining a latent variable graph generative model that allows for a flexible number of steps in encoding and sampling latent representations might thus prove a particularly suitable

approach to automatic chemical design. In the proposed architecture, I re-imagine a matrix of latent parameters as a canvas onto which we might consider attentively writing in a sequence of gradual refinement steps. In the decoder, I further consider latent samples as another canvas from which the model attentively reads different subpatches in conditioning each decoding step in the iterative generation process applied. With these modules in place, I hope to learn a latent representation of molecule space where the relative complexity of different molecules is better encoded and where a diverse set of samples from our latent distribution might produce a diverse set of interesting and valid molecule proposals.

Attentive Latent Writing and Reading Modules

In defining useful attentive writing and reading modules to incorporate into the neural network architecture, I must define attention models through which it is possible to differentiate. In Gregor et al.'s proposed DRAW model, the authors define such differentiable modules by means of an $N \times N$ grid of 2D Gaussian filters controlled by grid center parameters (g_X, g_Y), stride parameter δ, isotropic variance parameter σ^2, and a scalar intensity parameter γ. In the DRAW context, first intermediate representations of each of these parameters are obtained by a linear transformation of the current hidden state of the recurrent decoder module; that is,

$$(\tilde{g}_X, \tilde{g}_Y, \log \sigma^2, \log \tilde{\delta}, \log\gamma) = W(h^{dec}) \ . \tag{4.1}$$

The grid center parameters (g_X, g_Y) and stride parameter δ are then determined as

$$g_X = \frac{A+1}{2}(\tilde{g}_X + 1) \tag{4.2}$$

$$g_Y = \frac{B+1}{2}(\tilde{g}_Y + 1) \tag{4.3}$$

$$\delta = \frac{\max(A,B)-1}{N-1}\tilde{\delta} \tag{4.4}$$

for a given input image of dimensions $A \times B$.

The mean location of Gaussian filter (i, j) within the $N \times N$ grid is then determined by the following equations:

$$\mu_X^i = g_X + (i - N/2 - 0.5)\delta \tag{4.5}$$

$$\mu_Y^j = g_Y + (j - N/2 - 0.5)\delta \ . \tag{4.6}$$

Given these attention parameters, filterbank matrices F_X, F_Y are finally determined for each dimension as

$$F_X[i,a] = \frac{1}{Z_X}\exp\left(-\frac{(a-\mu_X^i)^2}{2\sigma^2}\right) \tag{4.7}$$

$$F_Y[j,b] = \frac{1}{Z_Y}\exp\left(-\frac{(b-\mu_Y^j)^2}{2\sigma^2}\right) \tag{4.8}$$

where (i,j) is a grid point in the filter grid, (a,b) is a pixel point in the input image, and Z_X, Z_Y are normalization constants ensuring that the summation over the corresponding input image dimension for each filterbank matrix is normalized to one.

Both attentive reading and writing are then operations largely defined by these filterbanks. In the original DRAW context, the attentive reading operation was defined as

$$read(x,\hat{x}_t,h_{t-1}^{dec}) = \gamma[F_Y x F_X^T, F_Y \hat{x} F_X^T] \tag{4.9}$$

where x is the current input image, $\hat{x}$ the error image or difference between the input image and current canvas image, and h_{t-1}^{dec} the previous hidden state of the recurrent decoder module. The attentive writing module, on the other hand, is further dependent on a writing patch w_t determined by a separate linear transformation of the current decoder hidden state. In formal terms, attentive writing is thus defined as

$$w_t = W'(h_t^{dec}) \tag{4.10}$$

$$write(h_t^{dec}) = \frac{1}{\hat{\gamma}}\hat{F}_Y^T w_t \hat{F}_X \tag{4.11}$$

where W' refers to a distinct weight matrix from W and $\hat{\gamma}, \hat{F}_X, \hat{F}_Y$ are used to refer to separately obtained Gaussian filter parameters from the ones obtained for the reading module. In this manner, the DRAW reading and writing modules define differentiable mechanisms by which we might attend to canvas subpatches of different positions, sizes, and resolutions.

In adapting these modules to writing out latent distribution parameters and reading latent samples, I make the following adjustments. For the writing module, the canvas is of dimension $N_z \times 2$, where N_z refers to the dimensionality of the latent space. In this manner, for each dimension of the latent space writing of the mean and variance parameters of the variational posterior is tied together. Parameters of the writing module are also now dependent on a recurrent module incorporated into the encoder architecture taking as input the output representations of a graph convolutional network architecture encoding the input batch of molecule samples. For the reading module, I remove the dependency on an "error

image" as this concept is not well-defined for the proposed application of the module. The input samples are also one-dimensional, thus this module now applies only a single filter F_X for reading. The reading module then outputs a "glimpse" of the latent samples for each generation step taken in the decoder.

4.2 Implementation and Training Details

The model is implemented in MXNet Chen et al. (2015), and builds off the open-source implementation of Li et al. (2018b) available at: github.com/kevinid/molecule_generator. For the purposes of proper comparison, I maintain many of the architectural specifications and hyperparameter settings reported by Li et al. (2018b). In particular, I build an encoder architecture based off of a six-layer graph convolutional network with layers of filter bank depth of 32, 64, 128, 128, 256, and 256. The activations from each of these graph convolutional layers are stacked together and passed to a pair of feedforward layers for outputting the latent distribution parameters. For architectures where the adapted DRAW writing module is incorporated, this initial prediction of the latent parameters was used as the canvas initialization that is then further refined by several rollout steps of the recurrent DRAW writing module. In architectures where I further incorporated the adapted DRAW reading module, the initial latent samples are passed to the recurrent DRAW reading module to produce a matrix of "glimpses" of different components of the latent samples for each generation step involved in reconstructing input molecules. Decoding then proceeds in the sequential generative process described previously. Generation step predictions are produced by a "policy" network of feedforward layers. Molecule initialization predictions are based off of a separate two-layer feedforward neural network conditioned on the latent samples produced from the encoder.

Due to constraints on computational resources and the time available for experiments, I was not able to train the different model architectures for the same number of epochs as Li et al. did in their experiments, nor was I able to perform extensive hyperparameter optimization. For the experiments on the ChEMBL dataset, I trained each of the proposed models for a single epoch with training batches of size 32. Training runs were performed on a single Nvidia Tesla K80 GPU, and the Adam (Kingma and Ba, 2014) optimization algorithm was used for training the network. I chose a latent dimension size of 100, and reading and writing window sizes of 20 for this latent dimension size. For the experiments on the QM9 and ZINC Drug-Like datasets, I used similar hyperparameter values and was able to train the full model for five epochs due to the significantly smaller sizes of these datasets. It should be noted that the results reported could likely be significantly improved with more capacity to search for optimal hyperparameter settings.

Chapter 5

Results

In this chapter, I discuss the empirical results obtained from training the proposed model on several standard molecule datasets. In particular, I study the performance of the model on a subset of the ChEMBL dataset up to 50 heavy atoms, the QM9 dataset, and a 250k sample from the ZINC Drug-Like dataset. The ChEMBL dataset includes approximately 1.5 million molecule samples, whereas the QM9 dataset and the ZINC Drug-Like dataset used include about 130k and 250k molecule samples respectively. Due to the considerably higher complexity of the ChEMBL dataset relative to the other benchmarks, I focus the experiments on this dataset and in particular perform a full ablation study of proposed model components on this benchmark.

To quantitatively evaluate the proposed model, I measure the percentage of valid, novel, and unique molecules produced by generating 1000 batches of 100 random samples after training. I evaluate whether molecules are valid using the open-source RDKit (Landrum, 2016) software package for cheminformatics and machine learning. I define the novelty metric as the ratio of the number of generated valid molecules that do not occur in the original dataset to the number of samples drawn for a given batch, and similarly define the uniqueness metric as the ratio of the size of the set of valid molecules generated to the total number of samples drawn. Thus both the novelty and uniqueness metrics implicitly incorporate molecular validity and numbers for these metrics should be upper-bounded by the validity results. I also evaluate and report the degree to which the proposed model is capable of capturing important molecular graph statistics of the studied datasets using metrics based upon those proposed by Liu et al. (2018).

5.1 ChEMBL Experiments

5.1.1 Ablation Study

Table 5.1 reports the results on ChEMBL incorporating different model components into the overall architecture. I obtain results for each ablated model while decoding with two different approaches: a probabilistic approach sampling at each turn a molecule generation step from a categorical distribution over the set of all possible append, connect, or end steps and a beam search-based approach aimed at determining the most probable decoding route that is computationally feasible to find. It should be noted that for reported beam search decoding results I generally only experimented with lower beam width values in the range of 1 to 5, and that reported results could likely be improved by experimenting with higher beam widths. As a shorthand, I refer to results obtained with probabilistic decoding and with beam search decoding by (p) and (bs) respectively.

I observe that, when decoding with beam search, each of the proposed model components leads to a significant improvement in the likelihood of generating valid molecule samples. However, in pushing for the model to produce molecular samples that are more and more likely to be valid, I find that optimizing for this metric can lead a significant drop in the diversity of samples generated, particularly when incorporating both the DRAW reading and writing module components. The full proposed model incorporating both DRAW modules does appear to lead to highly promising results, overall. Generating new molecules with this model and a stochastic decoding process leads to strong results across all three considered metrics, while sampling for the most likely decoding route with this model leads to a practical guarantee of generating valid molecules. More sophisticated decoding methods would likely lead to a model that can achieve the high likelihood of molecular validity desired while still producing a reasonably diverse set of novel molecule samples.

As is clearly evidenced by Table 5.1, one of the biggest issues in proposing a valuable generative model of molecular graphs is determining a model specification that best balances between producing proposals that are highly likely to be valid and proposals that are still well-differentiated from known molecules and likely to lead to new insights. I believe that defining a purely likelihood-based generative model that does not incorporate information on desirable chemical properties leads to a suboptimal model specification for best balancing between these objectives. The results from Table 5.1 could likely thus be improved by incorporating a property prediction module in a manner similar to previous works that helps with organizing the latent space for certain desirable properties and thus for conditioning the decoding process to be more likely to produce interesting molecules. Furthermore, incorporating more sophisticated techniques to remove some of the complexity in learning to

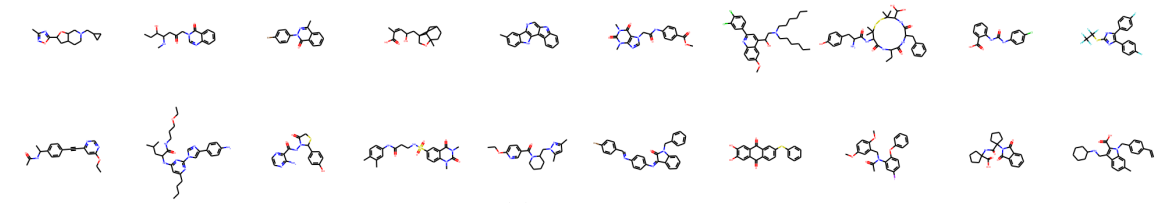
generate valid molecule samples would likely allow for the proposal of a generative model that can better focus on other metrics to optimize. In particular, I believe that the valency masking technique applied in Liu et al. (2018) would likely lead to significant improvements.

Table 5.1 Ablation study of proposed model components on ChEMBL.

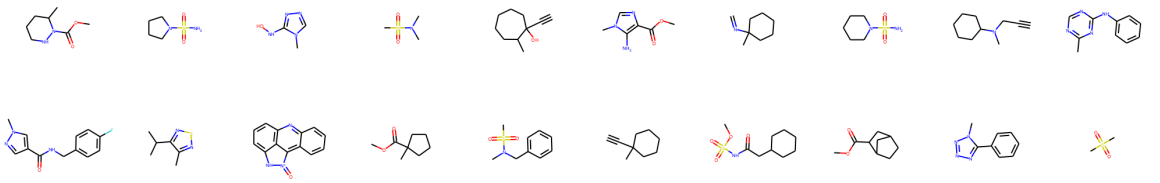
Model	% valid	% novel	% unique
VAE w/o DRAW (*p*)	0.933 ± 0.008	0.877 ± 0.011	0.836 ± 0.012
VAE w/o DRAW (*bs*)	0.982 ± 0.004	0.873 ± 0.009	0.545 ± 0.015
VAE + DRAW Writing (*p*)	0.957 ± 0.007	0.831 ± 0.012	0.633 ± 0.014
VAE + DRAW Writing (*bs*)	0.991 ± 0.003	0.825 ± 0.014	0.438 ± 0.015
VAE + DRAW Full (*p*)	0.956 ± 0.006	$\mathbf{0.942 \pm 0.007}$	$\mathbf{0.956 \pm 0.006}$
VAE + DRAW Full (*bs*)	$\mathbf{1.000 \pm 0.000}$	0.886 ± 0.008	0.007 ± 0.001

5.1.2 Generated Samples

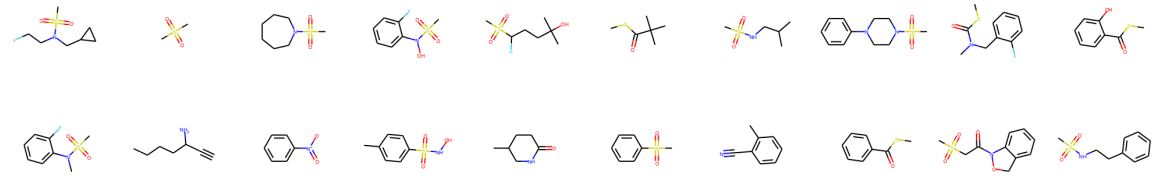
Fig. 5.1 shows random examples of molecule samples generated for each of the ablated models and compares these samples with random samples drawn from the original ChEMBL dataset. I observe that the proposed model without any DRAW components and with only the DRAW writing module incorporated both tend to be biased towards proposing relatively small and simple molecular proposals. It would appear that the attentive reading module is particularly important for biasing the model towards proposing relatively complex molecules. It is quite interesting that the different ablated models appear to have converged to different solutions for generating novel molecular samples. While all approaches achieve rather high validity results, it would appear that the space of possible solutions that optimize for this kind of metric is really quite large. I believe then that observed results could be further improved by incorporating additional constraints or objectives to prevent the model from converging to trivial solutions and biasing the model to converge to solutions that better align with the ultimate goals in automatic chemical design. In particular, while assessing the performance of the described model on the complex and diverse set of molecules involved in the ChEMBL dataset has its merits for validating the proof of concept, ultimately the "quality" of samples generated will depend significantly on more explicitly specifying the particular desirable properties for which we aim to optimize and better constraining or biasing the learned distributions to place probability mass primarily on regions corresponding to "interesting" molecule samples.



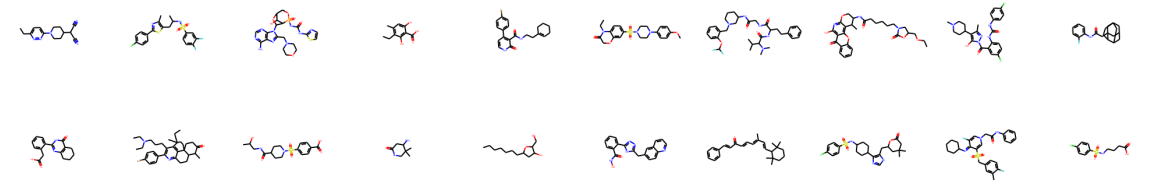
(a) ChEMBL.



(b) The proposed model without DRAW.



(c) The proposed model with DRAW Writing.



(d) The proposed model with both DRAW modules.

Fig. 5.1 Random selection of generated molecules for model variants trained on the ChEMBL dataset.

5.1.3 Model Comparison

Table 5.2 reports results comparing the full proposed model to the reported results for the two previous works, Li et al. (2018a) and Li et al. (2018b), applying similar molecular graph generative models to subsets of the ChEMBL dataset. Inspired by Liu et al.'s naming of the Li et al. (2018a) model as "deep graph autoregressive model" or "DeepGAR," I refer to the rather similar Li et al. (2018b) model as the "deep molecular graph autoregressive model" or "DeepMGAR." For convenience, I also refer to the full proposed model as "MGVAE" as a shorthand for "molecular graph VAE." For the purposes of this discussion, it should be noted that the DeepGAR model was trained on a much simpler subset of the ChEMBL dataset than the one used for these experiments, and thus it should be expected that the reported performance for this model should be higher.

I observe that the proposed model is highly competitive on ChEMBL with both of the previously proposed models. It is particularly promising that the full proposed model achieves performance comparable to the results reported for DeepMGAR despite considerably less training and limited hyperparameter tuning. Despite that the DeepGAR was trained and evaluated on a much simpler dataset than the one used here, MGVAE achieve comparable results to those reported in Li et al. (2018a). While neither of these previous studies reported metric evaluations for the diversity of samples generated, it remains highly promising that MGVAE is capable of producing just as many unique samples as it produces valid ones.

Table 5.2 ChEMBL model comparison results.

Model	% valid	% novel	% unique
DeepGAR* (Li et al., 2018a)	0.975	0.955	-
DeepMGAR (Li et al., 2018b)	0.955	0.985	-
MGVAE	0.952	0.930	0.952

5.1.4 Graph Statistics

Fig. 5.2 shows an empirical evaluation of the degree to which MGVAE captures important molecular graph statistics of the studied dataset, using a metric proposed by Liu et al. (2018). In particular, to determine the degree to which the model is able to capture the average atom- and bond-level statistics of molecules in the ChEMBL dataset, I compare the number of standard atom and bond types that appear in the average molecule from the original dataset to the same numbers for samples generated by MGVAE. I observe that the proposed model appears to well capture the fundamental atom- and bond-level statistics of the molecules in the ChEMBL dataset and appears able to regularly generate novel molecule proposals of similar composition and complexity to the actual molecules observed during training.

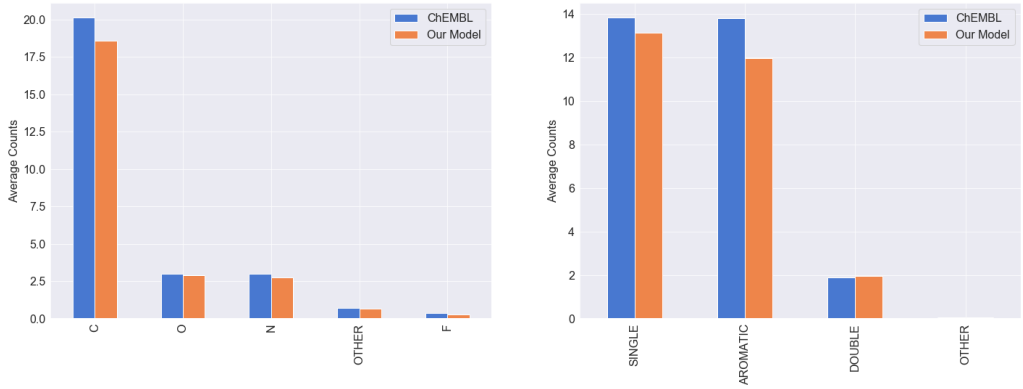


Fig. 5.2 Average molecular graph statistics for the ChEMBL dataset and the model generated samples.

5.2 QM9 Experiments

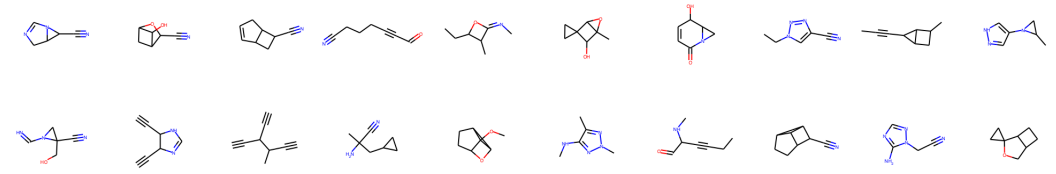
Table 5.3 reports results comparing MGVAE to previous work studying the relatively simple QM9 dataset. It should be noted that the results reported for Gómez-Bombarelli et al. (2016) and Kusner et al. (2017) are obtained from Simonovsky and Komodakis (2018) as the original papers do not report these metrics. Both in terms of the percentage of valid and novel molecules generated, MGVAE achieves a level of performance comparable to De Cao and Kipf's MolGAN or Liu et al.'s CGVAE. Furthermore, the proposed model significantly outperforms MolGAN in terms of the percentage of unique molecules generated, and makes for a considerable improvement over SMILES-based methods and Simonovsky and Komodakis's single-step graph generation approach.

In comparing MGVAE to previous work, it should be noted that due to time and computation constraints the proposed model was trained for only five epochs and hyperparameters were minimally tuned. I do, however, believe that the proposed model could stand to benefit from some of the domain specific techniques used in the MolGAN and CGVAE architectures. In particular, incorporation of a module for predicting whether latent samples decode to valid molecules similar to MolGAN's approach or directly making use of CGVAE's proposed valency masking technique would both likely lead to significant improvements.

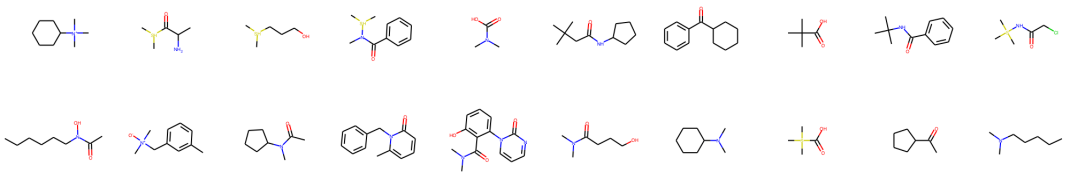
Table 5.3 QM9 model comparison results.

Model	% valid	% novel	% unique
ChemVAE (Gómez-Bombarelli et al., 2016)	0.103	0.900	0.675
GrammarVAE (Kusner et al., 2017)	0.602	0.809	0.093
GraphVAE (Simonovsky and Komodakis, 2018)	0.542	0.617	0.618
MolGAN (De Cao and Kipf, 2018)	0.998	0.981	0.032
CGVAE (Liu et al., 2018)	1.000	0.944	0.986
MGVAE	0.955	0.942	0.955

Fig. 5.3 shows random molecule samples generated from the full trained model along with random samples from the original dataset, and Fig. 5.4 reports a comparison of important molecular graph statistics between actual QM9 molecules and molecules generated by the trained model. I note that, in generating random samples, MGVAE does appear to have captured the relative complexity of molecules in the QM9 dataset and appears able to generate samples of complexity comparable to the particular dataset on which the model is trained. The strong correspondence in atom- and bond-level graph statistics between the generated samples and the original QM9 data samples further corroborates this conclusion.



(a) QM9 original data samples.



(b) MGVAE generated samples.

Fig. 5.3 Random selection of molecules for QM9 experiments.

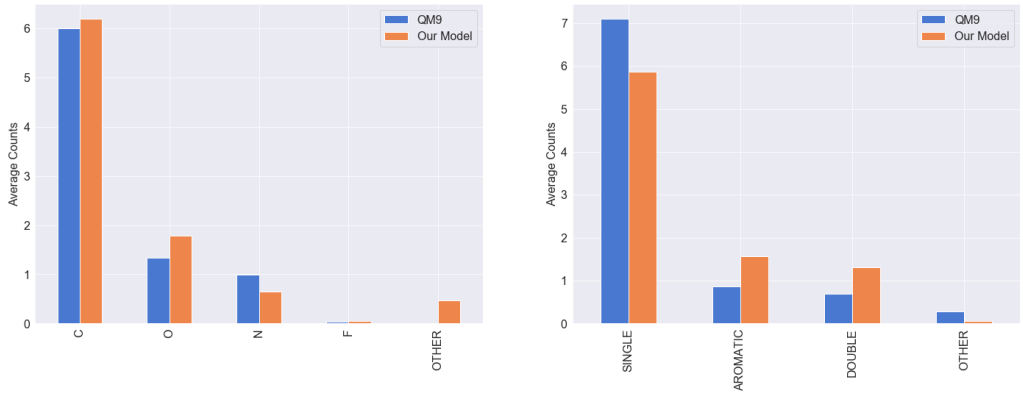


Fig. 5.4 Average molecular graph statistics for the QM9 dataset and the model generated samples.

5.3 ZINC Drug-Like Experiments

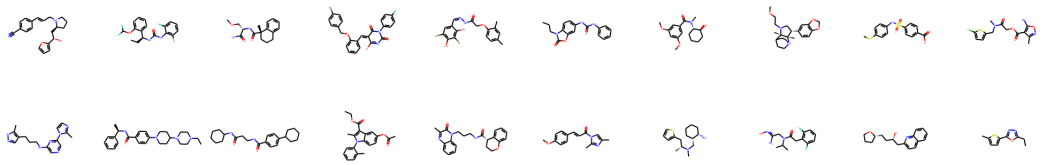
Table 5.4 reports results comparing MGVAE to previous work studying the ZINC Drug-Like dataset. It should be noted that the results reported for ChemVAE, GrammarVAE, and GraphVAE are obtained from Liu et al. (2018) as the original papers do not report these experimental results in studies on this particular dataset. Fig. 5.5 further shows random molecule samples generated by our full trained model along with random ZINC Drug-Like data samples, and Fig. 5.6 reports a comparison of important molecular graph statistics between the original ZINC Drug-Like dataset and the trained model.

Consistent with previous experiments, I observe that MGVAE significantly improves over SMILES-based methods and single-step graph generation methods, achieving an overall level of performance comparable with Liu et al.'s recently proposed CGVAE. While qualitative and quantitative evaluations seem to indicate that the model is again well capturing the particular

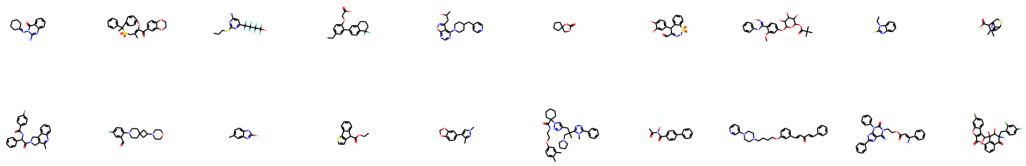
composition and relative complexity of ZINC Drug-Like molecules, it appears clear that the model could stand to be improved by incorporating more domain specific techniques for ensuring valid decoding and for encouraging a more informative structuring of the latent space. It should once again be noted that reported MGVAE results are based on a model trained for five epochs and with little hyperparameter tuning performed.

Table 5.4 ZINC Drug-Like model comparison results.

Model	% valid	% novel	% unique
ChemVAE (Gómez-Bombarelli et al., 2016)	0.170	0.980	0.310
GrammarVAE (Kusner et al., 2017)	0.310	1.000	0.108
GraphVAE (Simonovsky and Komodakis, 2018)	0.140	1.000	0.316
CGVAE (Liu et al., 2018)	1.000	1.000	0.996
MGVAE	0.943	0.932	0.943



(a) ZINC Drug-Like original data samples.



(b) MGVAE generated samples.

Fig. 5.5 Random selection of molecules for ZINC Drug-Like experiments.

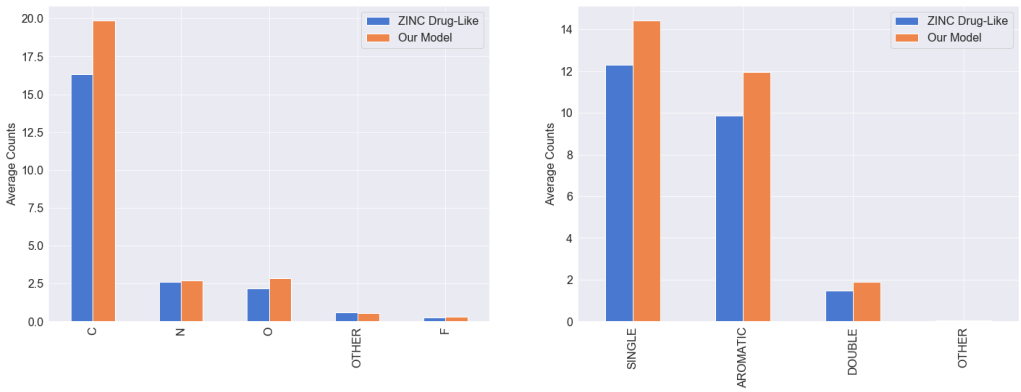


Fig. 5.6 Average molecular graph statistics for the ZINC Drug-Like dataset and the model generated samples.

5.4 Discussion

In discussing overarching takeaways from these results, I believe it would be best to recall the original stated purpose of the general line of research into automatic chemical design. In taking more of a machine learning-motivated approach to the problem of chemical design and drug discovery, the ultimate goal is to develop techniques that enable considerably more efficient search of the space of all synthesizable molecules that might prove to have intriguing properties in applications like medical treatment. I choose to approach this problem by means of adapting the general variational autoencoder framework for molecular graph generation. This basic model framework is particularly valuable for this problem in potentially enabling the learning of (1) a latent distribution for which sampling a new, diverse set of molecules after training can be done simply and efficiently, and (2) a structured latent space where axes of variation encode certain desirable properties for sampled molecules and where discovery of molecules optimized for a particular property can be performed (ideally) by simply traversing this latent space in some manner. If the promise of learning such a set of valuable tools holds, then this machine learning-based approach to chemical design could drastically improve the efficiency of the overall drug and molecule synthesis pipeline and potentially reshape the entire landscape of drug and material discovery, hopefully for the betterment of all.

It is in light of this viewpoint that I must analyze the results reported in this study, as well as the overall utility of evaluating potential solutions to this problem by means of the standard metrics generally reported in previous work. To what degree might the idealized versions of such models – as measured by the agreed upon metrics – ultimately prove useful in simplifying the entire pipeline of drug discovery, synthesis, and rigorous testing? For use in collaboration with biochemists and chemical engineers, we firstly would perhaps like to develop models that practically guarantee that any novel molecule samples proposed are, at minimum, chemically valid. However, as evidenced by my experimental results, optimizing this metric to the upper bound can lead to developing solutions that produce a much more constrained set of novel molecule proposals. In this manner, I note that aiming to guarantee validity of molecular samples implies an inherent tradeoff in the context of drug discovery: while we would like to aim primarily to learn models for which any novel samples generated are at minimum likely to be viable for further consideration, it would appear that at some point optimizing for this factor may overly constrain the space of novel molecules that would even be proposed for consideration.

The logical conclusion from this result would then appear to be that we should aim to learn latent representations of molecule space that "nearly" achieve a guarantee of producing valid molecule proposals while maintaining a high probability of producing novel and diverse proposals. However, in considering the ultimate value of a tool with such properties to a

biochemist, this argument breaks down. In selecting a single molecule proposal for further, highly costly development and testing, a model that proposes a large number of distinct molecule samples for small changes in variation of the input is just as unhelpful as one that always proposes the same molecule for large changes in variation. I thus argue that optimizing models for the novelty and diversity of samples produced misses the fundamental point of developing techniques for automatic chemical design.

In describing the problem in this manner, it might appear that the contention can be resolved simply by effective hyperparameter optimization, and by a proper definition of the prior and variational family of distributions in the context of variational autoencoders. However, I argue that any such design choice in this direction will suffer from framing the problem in this manner. Absent the context of a particular use case or at least of more biochemical motivation in defining the optimization objective, evaluating the standard generative properties of a generative model of molecules likely leads to the development of models improving performance in a direction that is not necessarily conducive to improvement on the original problem. Rather than aiming to directly optimize the measured diffuseness of a posterior over a learned latent space of molecules or to match the statistics of a dataset of molecules with little resemblance between distinct subsets, proposed models for the problem of automatic chemical design should aim to optimize and be evaluated upon metrics that more directly represent the ultimate use case for the idealized version of these tools.

Research on the problem of automatic chemical design and drug discovery has reached – or at least come very close to – the point where models can practically guarantee the validity of molecules proposed. Now that it has been demonstrated that the machine learning-based approach is capable of providing possible solutions, the next step is not to optimize and demonstrate the sheer capacity of all possible solutions such models might be able to generate. Rather, the next step is to work more in tandem with biochemists to determine better, more domain specific definitions of what properties of such generative models might be most beneficial to the entire pipeline involved in drug discovery.

Chapter 6

Conclusion

Rigorous, biochemically motivated drug discovery is an extremely challenging problem even for the most highly trained researchers. With a discrete, unstructured search space estimated to be about on the order of the number of atoms in the observable universe (Schneider and Fechner, 2005), exploring any substantial fraction of the space of synthesizable molecules is a daunting task for the entire biochemical field. Rather than try to approach this search problem naively, it likely would prove more fruitful to learn an alternative, continuous latent space representation of molecule space that well represents the particular chemical properties of interest in each of the axes of variation. In this manner, drug discovery ideally becomes a much simpler problem of moving to whichever region of the latent space best improves a set of desirable properties for a particular new drug proposal and decoding possible molecules by sampling from that region. In this work, I explore novel techniques for better learning and sampling from such a latent space representation of molecule space.

Building off recent work on the problem of automatic chemical design, I aim to propose a novel deep generative model architecture that, without many constrictive assumptions, achieves high likelihood of generating valid molecule samples while also aiming to make the samples generated as novel and diverse as possible. In particular, I propose a variational autoencoder model of molecular graphs that generates novel molecule samples by means of iteratively predicting a sequence of chemically motivated molecular generation steps given samples from the latent distribution. In order to better encode variation in molecular complexity into the latent parameters, I further propose the incorporation of an iterative attentive writing module for outputting the parameters of the latent distribution. I similarly propose the incorporation of an iterative attentive reading module for improving the model's ability to condition on different aspects of the latent samples at different steps in the molecule generation process. Here I show in experiments on subsets of the ChEMBL, QM9, and ZINC Drug-Like molecule datasets that the proposed model achieves strong results in terms of the

likelihood of generating valid new molecule proposals while demonstrating certain trade-offs in optimizing for molecular validity and for generating a diverse set of samples. I further observe interesting properties when incorporating into the architecture each of the proposed model components in terms of how each component biases the model towards different regions of the solution space. These results help further demonstrate that the latent variable deep generative model of graph approach is able to greatly mitigate many of the observed problems with previous approaches to automatic chemical design, and demonstrate that incorporating modules into the neural network architecture that help better encode variations in molecular complexity into the latent distribution is a promising direction of future research in ultimately tackling this general problem.

However, I contend that current metrics for evaluating the performance of generative models of molecules are poorly defined and not sufficiently domain specific. I argue that future directions for this research area should focus on optimizing metrics that better measure the ultimate value of such models in the drug discovery pipeline rather than more traditional metrics that measure the general capacity of any generative model. I hope that this work inspires further research on this problem incorporating more domain specific techniques in the molecule generation process and basing proposed solutions more explicitly on the particularities of modeling the data space of drug-like molecules.

References

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Dai, H., Tian, Y., Dai, B., Skiena, S., and Song, L. (2018). Syntax-directed variational autoencoder for structured data. *arXiv preprint arXiv:1802.08786*.

De Cao, N. and Kipf, T. (2018). Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*.

Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*.

Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. (2016). Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*.

Gori, M., Monfardini, G., and Scarselli, F. (2005). A new model for learning in graph domains. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 729–734. IEEE.

Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., and Wierstra, D. (2015). Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*.

Guimaraes, G. L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C., and Aspuru-Guzik, A. (2017). Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843*.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777.

Jin, W., Barzilay, R., and Jaakkola, T. (2018). Junction tree variational autoencoder for molecular graph generation. *arXiv preprint arXiv:1802.04364*.

Kang, S. and Cho, K. (2018). Conditional molecular design with deep generative models. *arXiv preprint arXiv:1805.00108*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kipf, T. N. and Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kusner, M. J., Paige, B., and Hernández-Lobato, J. M. (2017). Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*.

Landrum, G. (2016). Rdkit: Open-source cheminformatics software [online], version 2016.03. *Zenodo*.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. (2015). Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*.

Li, Y., Vinyals, O., Dyer, C., Pascanu, R., and Battaglia, P. (2018a). Learning deep generative models of graphs. *arXiv preprint arXiv:1803.03324*.

Li, Y., Zhang, L., and Liu, Z. (2018b). Multi-objective de novo drug design with conditional graph generative model. *arXiv preprint arXiv:1801.07299*.

Liu, Q., Allamanis, M., Brockschmidt, M., and Gaunt, A. L. (2018). Constrained graph variational autoencoders for molecule design. *arXiv preprint arXiv:1805.09076*.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Samanta, B., De, A., Ganguly, N., and Gomez-Rodriguez, M. (2018). Designing random graph models using variational autoencoders with applications to chemical design. *arXiv preprint arXiv:1802.05283*.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Schneider, G. and Fechner, U. (2005). Computer-based de novo design of drug-like molecules. *Nature Reviews Drug Discovery*, 4(8):649.

Simonovsky, M. and Komodakis, N. (2018). Graphvae: Towards generation of small graphs using variational autoencoders. *arXiv preprint arXiv:1802.03480*.

Weininger, D. (1988). Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36.

Williams, R. J. (1987). *Reinforcement-learning connectionist systems*. College of Computer Science, Northeastern University.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.