



저작자표시-동일조건변경허락 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.
- 이차적 저작물을 작성할 수 있습니다.
- 이 저작물을 영리 목적으로 이용할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



동일조건변경허락. 귀하가 이 저작물을 개작, 변형 또는 가공했을 경우에는, 이 저작물과 동일한 이용허락조건하에서만 배포할 수 있습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

이학석사학위논문

De Novo Drug Design Using Deep Generative Models

딥러닝 기반 생성 모델을 이용한 드 노보 약물 합성

2019년 2월

서울대학교 대학원

자연과학대학 통계학과

한 석 진

Abstract

Drug design seeks to generate chemical compounds aimed to satisfy specific preferable properties. In this paper, I propose a new drug design model called D4GAN, capable of producing molecule samples that match with a particular set of desirable metrics. This methodology combines recent advances in generative adversarial networks (GANs). I primarily used the ideas from boundary-seeking GAN (BSGAN) and objective-reinforced GAN (ORGAN), to directly deal with molecules encoded as text sequences. I also adopt Wasserstein GAN (WGAN), to improve the convergence and quality of generated *de novo* drug candidates. Moreover, I adopt variational autoencoder (VAE) for the generator in GAN, in order to improve the stability and quality of sample model generation, avoiding mode collapse that often happens in other GAN models. The results show that D4GAN successfully tunes the structure and quality of generated samples.

Keywords: Drug design, Generative model, Deep learning

Student Number: 2017-21920

Contents

Abstract	i
Contents	iii
List of Figures	v
List of Tables	vii
1 Introduction	1
2 Review of Generative Models	5
2.1 Variational Autoencoder (VAE)	5
2.2 Generative Adversarial Network (GAN)	6
2.2.1 GAN with Discrete Data	6
2.3 VAE/GAN	7
3 D4GAN: Discrete <i>De Novo</i> Drug Design Using GAN	9
3.1 Overview of Training Algorithm	9
4 Experiments	11
4.1 Details on Experimental Setup	11
4.1.1 Encoder	11

4.1.2	Generator	12
4.1.3	Discriminator	12
4.1.4	Training Set	13
4.1.5	Evaluating Organic Compounds	14
4.1.6	Other Information	14
4.2	Results	15
4.2.1	Chemical Properties	15
4.2.2	Examples	16
5	Conclusions	19
	Bibliography	21
	초 록	25
	Acknowledgments	27

List of Figures

2.1	The overall structure of VAE/GAN (Larsen et al., 2015)	8
4.1	Structure of the encoder	12
4.2	Structure of the generator	13
4.3	Structure of the discriminator	13
4.4	Learning curves of D4GAN	15
4.5	Violin plots of training samples and generated samples	16
4.6	Examples of well-known organic compounds found in generated samples	17
4.7	Examples of unknown organic compounds found in generated samples .	17

List of Tables

1.1	Examples of SMILES representation	3
4.1	Summary statistics among generated valid samples	16

Chapter 1

Introduction

Over the last decades, the discovery and development of novel drug candidates have been a challenging issue in the pharmaceutical industry. The cost of developing new drugs has been soaring nowadays. Recent work shows that the estimated cost of developing a single cancer drug was 648 million dollars, and the launching price for a new drug has doubled over the past two decades. Failures in the development process are critical contributors to this cost. The estimated clinical approval success rate is only 11.8% (Prasad and Mailankody, 2017).


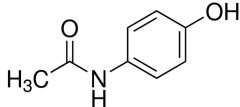
The very beginning step of new drug developing process, the prehuman/preclinical step, where researchers search for potential drug candidates from large molecule database, takes its share about 28.6% in the total cost of a drug development trial (Avorn, 2015). Definitely, this pilot study is one of the biggest drivers of the cost in pharmaceutical industrial innovation aside for Phase III stage. Recently, massively stacking datasets on molecular structures and their biochemical properties make this process more efficient and comprehensive. However, at the same time, demand for more effective methods that guarantee both diversity and druglikeness of potential molecular candidates is a rising concern (Roy, 2012).

New generative methods in deep learning have gained interest in computer-assisted *de novo* drug design (CADD). In the mainstream, methods using qualitative structure activity relationship (QSAR) or molecular docking have been suggested in the new drug design (Cherkasov et al., 2014; Meng et al., 2011). These methods associate particular molecular structures or ligands to the chemical properties of the molecule with several data analysis techniques, such as linear regressions, support vector machines, stochastic tree searches. However, because of too much complexity and variability of drug-like molecule family, the methods using statistics, combinatorics or machine learning have had limited ability to analyze and generate potential candidates thoroughly.

Recently, with the advent of deep neural network techniques and parallel computing, researchers have enjoyed more complicated non-linear models on the design of novel molecules. Models using neural networks like long short-term memory (LSTM) (Segler et al., 2017), variational autoencoder (VAE) (Blaschke et al., 2017) or reinforcement learning-based generative adversarial network (GAN) (Guimaraes et al., 2017) have been suggested, and the reinforcement learning-based language model has also been adopted in generating sample molecules (Olivecrona et al., 2017). These model think of molecules as a sequence of atoms, so used SMILES (simplified molecular-input line-entry system; see Table 1.1 for examples) for describing the grammar of how the atoms are composed to build the structure of the given molecule. Another state-of-the-art model, which think of molecules as a sequence of chemical reactions, adopted Monte Carlo tree search (MCTS) also showed compatible potential for generating new molecules (Segler et al., 2018).

In this paper, I propose a new drug design model called D4GAN, capable of producing molecule samples that match with a particular set of desirable metrics. This methodology combines recent advances in GANs. I primarily used the ideas

Table 1.1: Examples of SMILES representation

Chemical Name	Structure	SMILES formula
Benzene		<chem>C1=CC=CC=C1</chem>
Acetaminophen		<chem>CC(=O)NC1=CC=C(C=C1)O</chem>

from boundary-seeking GAN (BSGAN) (Hjelm et al., 2018) and objective-reinforced GAN (ORGAN) (Guimaraes et al., 2017). Both models are designed for training a generative model adversarially with discrete data, which is expected to directly deal with the discrete sequential nature of SMILES dataset in a stable manner. I also adopt Wasserstein GAN (WGAN), which uses Wasserstein distance for assessing the loss of the generated sample, to improve the convergence and quality of generated *de novo* drug candidates (Arjovsky et al., 2017). Moreover, I adopt VAE for the generator in GAN, in order to improve the stability and quality of sample model generation, avoiding the mode collapse that often happens in other GAN models (Larsen et al., 2015).

I test the model in the context of drug generation, optimizing several molecule metrics. The results show that D4GAN successfully tunes the structure and quality of generated samples.

Chapter 2

Review of Generative Models

In this section, I elaborate on variational autoencoders (VAEs) and generative adversarial networks (GANs) in the context of probability measures. Then, I introduce GAN with discrete data and VAE/GAN, both of which provide a fundamental background for D4GAN.

2.1 Variational Autoencoder (VAE)

A VAE (Kingma and Welling, 2014) consists of two networks called an *encoder* and a *decoder*. An encoder maps a data sample $x \in X$ to a probability measure on a latent feature space \mathbb{R}^P , and a decoder maps the latent feature $z \in \mathbb{R}^P$ back to its corresponding probability measure on X . In the case where X is finite (*i.e.*, x is discrete), one can put the encoder and decoder as specified below:

$$z \sim \text{Enc}_\phi(x) \equiv \mathcal{N}(\mu_\phi(x), \text{diag}(\sigma_\phi^2(x))), \quad x \sim \text{Dec}_\theta(z) \equiv \text{Softmax}(p_\theta(z)).$$

Training a VAE can be conducted by optimizing ELBO (evidence lower bound), the sum of negative expected log likelihood and the Kullback-Leibler divergence (KL

divergence):

$$\mathcal{L}_{VAE} = -\mathbb{E}_{\text{Enc}_\phi(x)}[\log \text{Dec}_\theta(u)] + D_{KL}(\text{Enc}_\phi(x) \parallel \mathcal{N}(0, I)).$$

Note that the negative expected log likelihood can be viewed as a reconstruction loss, and the KL divergence can be considered as a regularizer.

2.2 Generative Adversarial Network (GAN)

A GAN (Goodfellow et al., 2014) is comprised of two networks: a *generator* and a *discriminator*. The generator \mathbb{Q}_ψ maps latent $z \in \mathbb{R}^P$ to a probability measure on data space X . The discriminator D_θ classifies whether the given data $x \in X$ is chosen from a real dataset or from a dataset produced by the generator. After training, one can expect the generator to produce high-quality samples indistinguishable from real data. Training a GAN is performed by minimizing

$$\mathcal{V}(\mathbb{P}, \mathbb{Q}_\psi, D_\theta) = \mathbb{E}_{\mathbb{P}}[\log D_\theta(x)] + \mathbb{E}_{h(z)}[\log(1 - D_\theta(G_\psi(z)))],$$

where optimizer of D_θ aims to maximize $\mathcal{V}(\mathbb{P}, \mathbb{Q}_\psi, D_\theta)$ and optimizer of G_ψ tends to minimize it.

In practice, however, the GAN struggles to achieve stable development and training. Therefore, variants of this model were developed as an alternative. The most popular among these is WGAN (Arjovsky et al., 2017), which adopts Wasserstein distance in the objective function of GAN: $\mathcal{V}(\mathbb{P}, \mathbb{Q}_\psi, D_\theta) = \mathbb{E}_{\mathbb{P}}[D_\theta(x)] - \mathbb{E}_{h(z)}[D_\theta(G_\psi(z))]$.

2.2.1 GAN with Discrete Data

In GANs, optimizing the value function $\mathcal{V}(\mathbb{P}, \mathbb{Q}_\psi, D_\theta)$ is only possible when the generated samples are completely differentiable with respect to θ . For discrete data, the sampling process is not differentiable, and thus several variations of GAN exist

to deal with such data type. One popular variation is SeqGAN (Yu et al., 2017). A SeqGAN model trains the generator \mathbb{Q}_ψ as an agent in a reinforcement learning context. Instead of minimizing $\mathcal{V}(\mathcal{P}, \mathbb{Q}_\psi, D_\theta)$, this model maximizes the expected long-term reward, where the reward function is given by $R(X) = D_\theta(X)$. Building on the ideas of SeqGAN, ORGAN (Guimaraes et al., 2017) takes into account additional molecule metrics O . The reward function is extended to a linear combination of D_θ and O , parametrized by λ : $R(X) = \lambda D_\theta(X) + (1 - \lambda)O(X)$. Note that in both SeqGAN and ORGAN, the policy gradient can be computed using the REINFORCE algorithm (Williams, 1992).

BSGAN (Hjelm et al., 2018), in the vein of f -GAN (Nowozin et al., 2016), optimizes the variational lower bound for the f -divergence, $\mathcal{V}(\mathbb{P}, \mathbb{Q}_\psi, T_\theta) = \mathbb{E}_{\mathbb{P}}[\nu \circ F_\theta(x)] - \mathbb{E}_{\mathbb{Q}_\psi}[f^*(\nu \circ F_\theta(x))]$ where the generator is given by $T_\theta = \nu \circ F_\theta$. It computes target generated samples density from f -divergence *importance weights*,

$$\tilde{p}(x) = \frac{(\partial f^*/\partial T)T(x)}{\mathbb{E}_{\mathbb{Q}_\psi}[(\partial f^*/\partial T)T(x)]} q_\psi(x),$$

then provides a policy gradient for training the generator when the empirical density $q_\psi(x)$ is discrete. Lower-variance version of this policy gradient is obtained in the form of expected conditional KL divergence or reverse KL divergence, which can be computed using Monte Carlo estimation or the REINFORCE algorithm.

2.3 VAE/GAN

A VAE/GAN (Larsen et al., 2015) attempts to combine the advantage of GAN as a high-quality generative model and VAE as a method that produces an encoder of data into the latent feature space \mathbb{R}^P . It trains the VAE and GAN jointly, so as to make the generator learn the advantages of both VAE and GAN. Then, the generator can reproduce a molecule from the encoded feature, and produce a new result from an

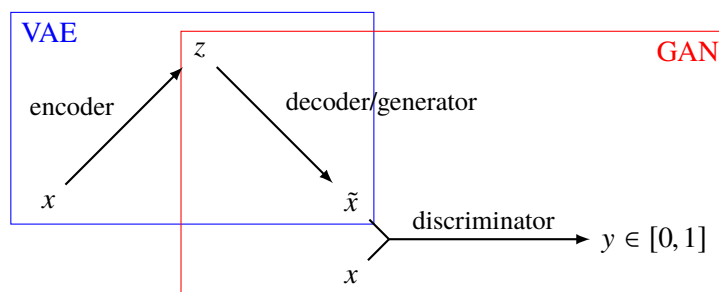


Figure 2.1: The overall structure of VAE/GAN (Larsen et al., 2015)

arbitrary element in latent feature space \mathbb{R}^p . Figure 2.1 illustrates the main idea of VAE/GAN.

Chapter 3

D4GAN: Discrete *De Novo* Drug Design Using GAN

3.1 Overview of Training Algorithm

Algorithm 3.1 provides an outline of the training procedure of D4GAN.

The overall structure of D4GAN is analogous to VAE/GAN. The significant difference between D4GAN and VAE/GAN is the loss functions. D4GAN applies Wasserstein loss, BSGAN and ORGAN style loss functions, which are not included in VAE/GAN. The VAE/GAN model serves as a mechanism to prevent mode collapse for D4GAN. In practice, applying only BSGAN and ORGAN models with Wasserstein loss results in unstable learning and frequent mode collapse. Furthermore, VAE/GAN and D4GAN maneuver distinct data types: image data and sequential data, respectively. Thus, while VAE/GAN replaces the VAE reconstruction (expected log likelihood) error term with a reconstruction error expressed in the GAN discriminator, D4GAN does not change the reconstruction loss.

Algorithm 3.1 Training D4GAN

Require: Encoder network, $\text{Enc}_{\theta_E} : \Sigma^T \rightarrow \mathbb{R}^p \times \mathbb{R}^p$

Require: Generator network, $\text{Gen}_{\theta_G} : \mathbb{R}^p \rightarrow \mathcal{M}(\Sigma^T)$

Require: Discriminator network, $\text{Dis}_{\theta_D} : \Sigma^T \rightarrow [0, 1]$

- 1: Initialize parameters $\theta_E, \theta_G, \theta_D$
 - 2: **repeat**
 - 3: Pick $X \in \Sigma^T$ randomly from the training set
 - 4: $\mu, \sigma^2 \leftarrow \text{Enc}_{\theta_E}(X)$
 - 5: $\mathcal{L}_{KL} \leftarrow \frac{1}{2} \sum_{k=1}^p \left(\mu_k^2 + \sigma_k^2 - 2 \log \sigma_k - 1 \right)$ ▷ KL divergence as regularizer
 - 6: $\tilde{g} \leftarrow \text{Gen}_{\theta_G}(\tilde{Z})$ where $\tilde{Z} \sim N(\mu, \text{diag}(\sigma^2))$, $\hat{g} \leftarrow \text{Gen}_{\theta_G}(\hat{Z})$ where $\hat{Z} \sim N(0, I)$
 - 7: $\tilde{X} \sim \tilde{g}$, $\hat{X}_j \sim i.i.d. \hat{g}$ for $j = 1, 2, \dots, m$
 - 8: $\mathcal{L}_{Recon} \leftarrow -\log \tilde{g}(X)$ ▷ Negative log-likelihood loss as the reconstruction loss
 - 9: $\omega_j \leftarrow \frac{\text{Dis}_{\theta_D}(\hat{X}_j)}{1 - \text{Dis}_{\theta_D}(\tilde{X}_j)}$ for $j = 1, 2, \dots, m$
 - 10: $\mathcal{L}_{RL} \leftarrow -\sum_{j=1}^m \left[\lambda O(\hat{X}_j) + (1 - \lambda) \frac{\omega_j}{\sum_{k=1}^m \omega_k} \right] \log \hat{g}(\hat{X}_j)$ ▷ ORGAN + BSGAN loss
 - 11: $\mathcal{L}_{GAN} \leftarrow -\left[\text{Dis}_{\theta_D}(X) - \text{Dis}_{\theta_D}(\tilde{X}) - \frac{1}{m} \sum_{j=1}^m \text{Dis}_{\theta_D}(\hat{X}_j) \right]$ ▷ Wasserstein loss
 - 12: $\theta_E \leftarrow \theta_E - \eta_E \nabla_{\theta_E} (\mathcal{L}_{KL} + \mathcal{L}_{Recon})$
 - 13: $\theta_G \leftarrow \theta_G - \eta_G \nabla_{\theta_G} (\kappa \mathcal{L}_{Recon} + \mathcal{L}_{RL})$
 - 14: $\theta_D \leftarrow \theta_D - \eta_D \nabla_{\theta_D} \mathcal{L}_{GAN}$
 - 15: **until** converge
-

Chapter 4

Experiments

4.1 Details on Experimental Setup

Here I test the effectiveness of D4GAN for generating molecules with advantageous properties in the context of drug design. Here, molecules are encoded text sequences by using the SMILES representation of a molecule.

4.1.1 Encoder

Figure 4.1 depicts the structure of the encoder network. The encoder maps a given SMILES sequence $(X_1, X_2, \dots, X_T) \in \Sigma^T$ into a feature vector $z \in \mathbb{R}^P$ by sampling z from a multivariate normal distribution $N(\mu, \text{diag}(\sigma^2))$. I apply a simple gated recurrent unit (GRU) (Cho et al., 2014) after an embedding layer, where the dimension of the embedded vector is set to be 32. Note that only the hidden layer of GRUs is used and the outputs are ignored. I then apply two distinct fully-connected layers to obtain $\mu, \log \sigma \in \mathbb{R}^P$. The dimension of the feature space and hidden layer of GRUs is both chosen to be 512.

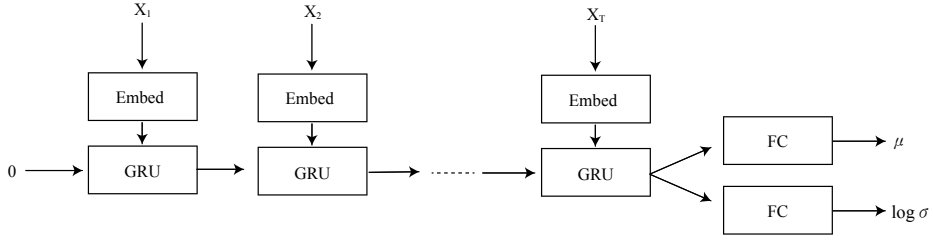


Figure 4.1: Structure of the encoder

4.1.2 Generator

Figure 4.2 clarifies the structure of the generator network. The generator produces SMILES sequences as follows. At time t , the generator returns new features $z_t \in \mathbb{R}^p$ and a probability measure g_t , given previous features $z_{t-1} \in \mathbb{R}^p$ and previous SMILES characters $x_{t-1} \in \Sigma$. Next, the generator samples x_t from g_t . Here an embedding layer and a GRU are used again as in the encoder network, with fully-connected layers and softmax layers used to construct a probability measure. The dimension of the embedding layer and output layer of GRU is 32 and 512, respectively. In addition, I employ teacher forcing (Williams and Zipser, 1989) with probability 0.9, when training the generator.

4.1.3 Discriminator

Figure 4.3 denotes the structure of the discriminator network. The discriminator classifies the given SMILES sequences as real or generated. The discriminator is a convolutional neural network. All leaky ReLU layers have a negative slope of 0.2, and all convolutional layers have output channels of 128, a kernel size of 5, stride of 1 and padding of 2.

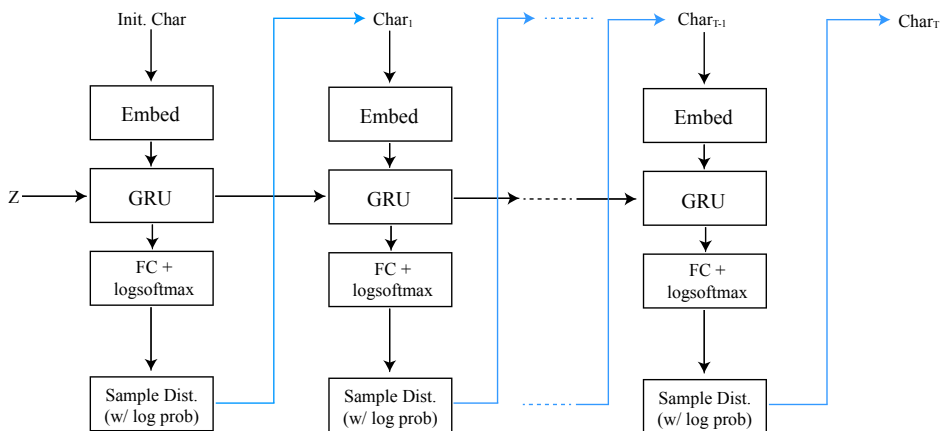


Figure 4.2: Structure of the generator

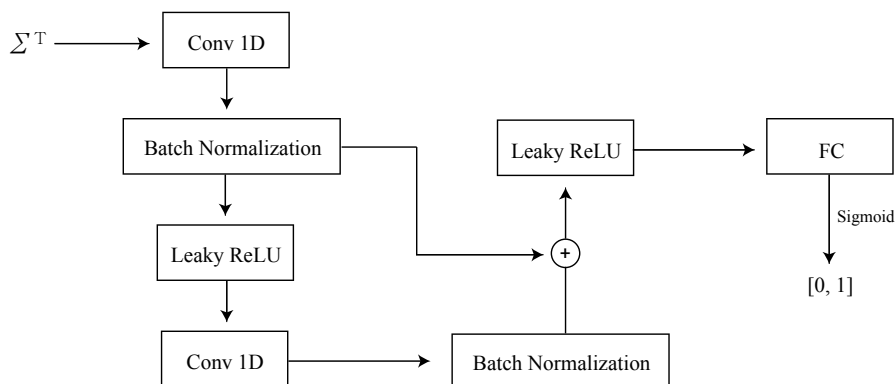


Figure 4.3: Structure of the discriminator

4.1.4 Training Set

For training, I follow the setting of Guimaraes et al. (2017). I utilized a random subset of 5,000 molecules from the set of 134 thousand stable small molecules. I regard each molecule’s SMILES code as a multinomial Bernoulli distribution sequence. The maximum sequence length is 98 and the size of alphabets is 41.

4.1.5 Evaluating Organic Compounds

Evaluating the validity and quality of the generated samples is essential. I judge the generated molecules based on the following *in silico* chemical properties. I picked qualities that are normally desired for small molecule discovery.

- Solubility: a property that measures how likely a molecule can mix with water, also known as the water-octanol partition coefficient ($\log P$) (Wildman and Crippen, 1999).
- Synthesizability: estimates how hard (0) or easy (1) how it is to synthesize a given molecule (Ertl and Schuffenhauer, 2009).
- Druglikeness: how likely a molecule is a viable candidate for a drug, an estimate that captures the abstract notion of aesthetics in medicinal chemistry (Bickerton et al., 2012).

Molecule metrics are implemented using the RDKit chemoinformatics package¹. I referred to the code for ORGAN, including metrics for each experiment, available on GitHub to evaluate these properties².

4.1.6 Other Information

I used Adam (Kingma and Ba, 2014) to train D4GAN, the learning rates of the encoder and generator set to be 10^{-3} and 10^{-5} for the discriminator. Also, I chose $\lambda = 0.5$ and $\kappa = 4$, where λ and κ are as given in Algorithm 3.1. The model is trained with 100 epochs with a mini-batch size of 64. I developed D4GAN using PyTorch and RDKit.

¹<https://www.rdkit.org/>

²<https://github.com/gablg1/ORGAN>

4.2 Results

I generated 5,000 molecules using D4GAN. Among them, 766 were valid, and 620 were chosen as final candidates after removing duplicates. Figure 4.4 shows the learning curves of D4GAN with respect to different molecule metrics. The plots are based on mean values of valid generated molecules, except for validity. Note that the plots are not drawn at the initial stages since there is no valid data at that point.

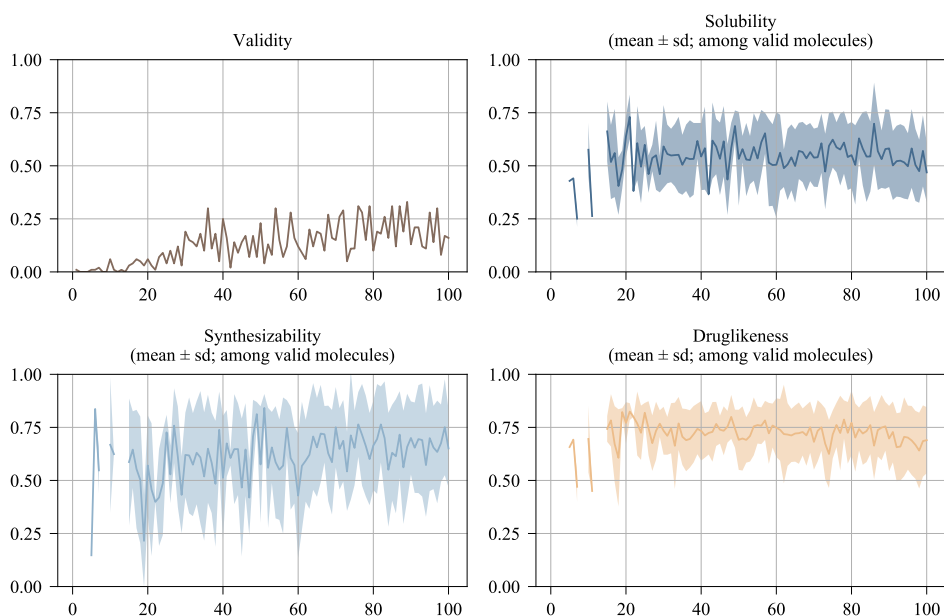


Figure 4.4: Learning curves of D4GAN

4.2.1 Chemical Properties

Table 4.1 shows summary statistics for evaluation of chemical properties. Figure 4.5 displays violin plots of chemical properties in the training set and the generated set. Both show that D4GAN is able to capture successfully mimics the distribution of properties of the training set well.

Table 4.1: Summary statistics among generated valid samples

Solubility				
Mean: 0.547		SD: 0.161		
min: 0.000	Q1: 0.451	Q2: 0.556	Q3: 0.657	max: 1.000
Synthesizability				
Mean: 0.651		SD: 0.246		
min: 0.000	Q1: 0.504	Q2: 0.701	Q3: 0.834	max: 1.000
Druglikeness				
Mean: 0.720		SD: 0.146		
min: 0.274	Q1: 0.623	Q2: 0.751	Q3: 0.840	max: 0.944

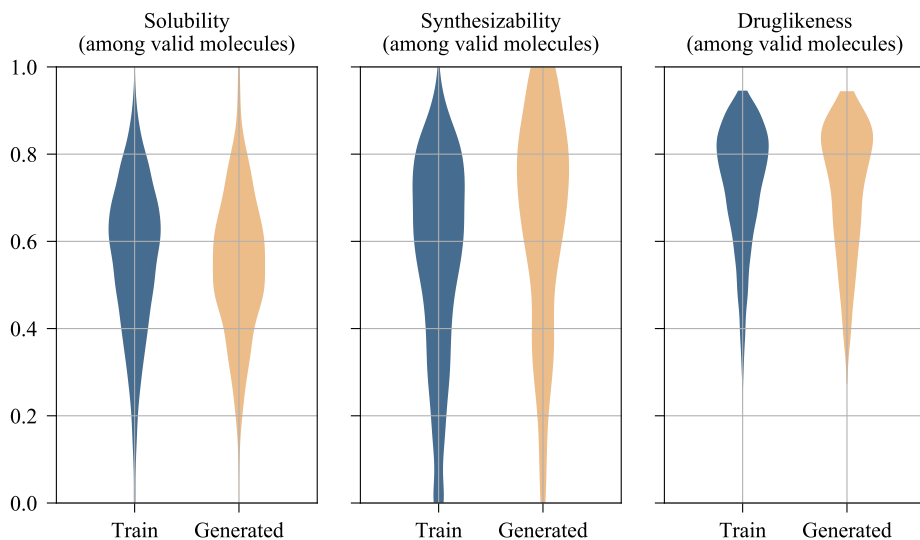


Figure 4.5: Violin plots of training samples and generated samples

4.2.2 Examples

Figure 4.6 and 4.7 show molecules generated using D4GAN. It should be noted that I was able to create molecules that are *not* in the initial training set.

Well-known Organic Compounds

Among the 620 selected molecules around 20 of them were familiar organic compounds, for instance, ethanol, formaldehyde, methylamine, and methylpyridine. Methylamine and methylpyridine are the primary precursors to caffeine and vitamin B, respectively.

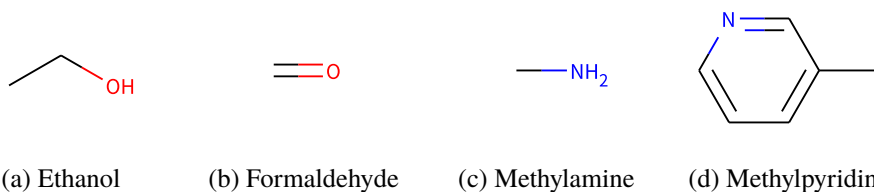


Figure 4.6: Examples of well-known organic compounds found in generated samples

Unknown Organic Compounds

A large portion of generated samples were unknown organic compounds. I could not find these compounds in PubChem, a database of chemical molecules and their activities against biological assays.

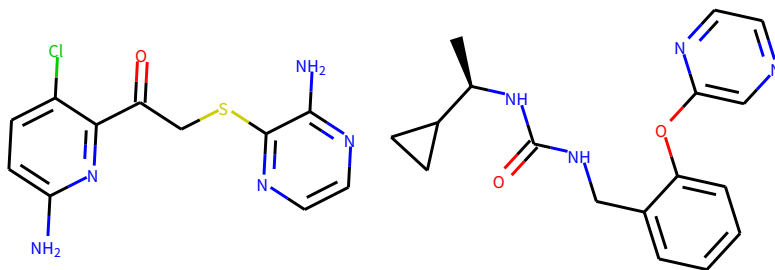


Figure 4.7: Examples of unknown organic compounds found in generated samples

Chapter 5

Conclusions

In this work, I have presented D4GAN, a new framework for generating organic compounds with desirable properties in a pharmaceutical context of drug discovery. I have built on basic structures of ORGAN, BSGAN, and WGAN, and extended them with VAE/GAN to handle mode collapse.

I have shown that D4GAN can improve certain metrics that desired, namely solubility, synthesizability, and druglikeness. More importantly, non-repetitive samples were generated, a large portion of which were novel organic compounds not registered in any public chemical substances database. Chemical properties of the generated data show its competency to be of practical use in the field of organic chemistry as well as the pharmaceutical industry.

One area of improvement is to complicate D4GAN's current network framework. Tuning model hyperparameters and adopting different molecule metrics are one of many possibilities. I believe these extensions to be quite promising since the present structure is largely simple. Furthermore, at present D4GAN uses publicly available datasets for training, which does not contain sufficient enough likely candidates for drug discovery. Forthcoming research should diversify chemical compounds in the

training sets to promote diversity of generated samples.

Bibliography

- Arjovsky, M., S. Chintala, and L. Bottou (2017). Wasserstein GAN. *arXiv preprint arXiv:1701.07875v3*.
- Avorn, J. (2015). The \$2.6 billion pill – methodologic and policy considerations. *New England Journal of Medicine* 372(20), 1877–1879.
- Bickerton, G. R., G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins (2012). Quantifying the chemical beauty of drugs. *Nature chemistry* 4(2), 90.
- Blaschke, T., M. Olivecrona, O. Engkvist, J. Bajorath, and H. Chen (2017). Application of generative autoencoder in de novo molecular design. *Molecular informatics*.
- Cherkasov, A., E. N. Muratov, D. Fourches, A. Varnek, I. I. Baskin, M. Cronin, J. Dear-den, P. Gramatica, Y. C. Martin, R. Todeschini, et al. (2014). Qsar modeling: where have you been? where are you going to? *Journal of medicinal chemistry* 57(12), 4977–5010.
- Cho, K., B. Van Merriënboer, D. Bahdanau, and Y. Bengio (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

- Ertl, P. and A. Schuffenhauer (2009). Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics* 1(1), 8.
- Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680.
- Guimaraes, G. L., B. Sanchez-Lengeling, C. Outeiral, P. L. C. Farias, and A. Aspuru-Guzik (2017). Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843*.
- Hjelm, R. D., A. P. Jacob, T. Che, A. Trischler, K. Cho, and Y. Bengio (2018). Boundary-seeking generative adversarial networks. *arXiv preprint arXiv:1702.08431v4*.
- Kingma, D. P. and J. Ba (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Kingma, D. P. and M. Welling (2014). Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114v10*.
- Larsen, A. B. L., S. K. Sønderby, H. Larochelle, and O. Winther (2015). Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300v2*.
- Meng, X.-Y., H.-X. Zhang, M. Mezei, and M. Cui (2011). Molecular docking: a powerful approach for structure-based drug discovery. *Current computer-aided drug design* 7(2), 146–157.

- Nowozin, S., B. Cseke, and R. Tomioka (2016). f -GAN: Training generative neural samplers using variational divergence minimization. *arXiv preprint arXiv:1606.00709v1*.
- Olivecrona, M., T. Blaschke, O. Engkvist, and H. Chen (2017). Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics* 9(1), 48.
- Prasad, V. and S. Mailankody (2017). Research and development spending to bring a single cancer drug to market and revenues after approval. *JAMA internal medicine* 177(11), 1569–1575.
- Roy, A. S. (2012). Stifling new cures: the true cost of lengthy clinical drug trials. *Manhattan Institute for Policy Research* 5, 5–13.
- Segler, M. H., T. Kogej, C. Tyrchan, and M. P. Waller (2017). Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS Central Science*.
- Segler, M. H., M. Preuss, and M. P. Waller (2018). Planning chemical syntheses with deep neural networks and symbolic ai. *Nature* 555(7698), 604.
- Wildman, S. A. and G. M. Crippen (1999). Prediction of physicochemical parameters by atomic contributions. *Journal of chemical information and computer sciences* 39(5), 868–873.
- Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8, 229–256.
- Williams, R. J. and D. Zipser (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural computation* 1(2), 270–280.

Yu, L., W. Zhang, J. Wang, and Y. Yu (2017). SeqGAN: Sequence generative adversarial nets with policy gradient. *arXiv preprint arXiv:1609.05473v6*.

초 록

신약 후보군 디자인 단계에서는 약물이 가져야 할 특정한 화학적 성질들을 만족하는 화합물을 생성하는 것을 목표로 한다. 본 논문에서 새로 제안한 모형인 D4GAN을 이용하면 특정한 지표 수준을 만족하는 화합물 샘플들을 생성할 수 있다. D4GAN은 최근 생산적 적대 신경망 분야에서 개발된 여러 모형을 결합하였다. D4GAN의 핵심적인 구조로 boundary-seeking GAN (BSGAN) 및 objective-reinforced GAN (ORGAN)을 사용하였는데, 이들은 문자열의 형태로 인코딩된 화합물 데이터를 직접적으로 다루기 위한 생산적 적대 신경망의 변형이다. 또한, D4GAN이 생성하는 드 노보 약물 후보 물질들의 수렴 및 품질 개선을 위해 Wasserstein GAN (WGAN)을 도입하였다. 덧붙여, GAN의 생성자에 variational autoencoder (VAE)를 결합하여, GAN에서 종종 발생하는 mode collapsing 문제를 피하고 생성된 화합물 샘플들의 안정성과 품질을 개선하고자 하였다. 간단한 실험 결과, D4GAN이 생성한 약물 후보군 샘플들의 품질과 구조가 성공적으로 조율되었음을 확인하였다.

주요어: 약물 합성, 생성 모형, 딥러닝

학번: 2017-21920

Acknowledgments

This work is originally presented as a team project of the class 326.739A in 2018 spring semester. I would like to express a great appreciation to my teammates—Hyoshin Kim, Kyeongwon Lee, and Seowon Choi—for allowing me to publish this work as my dissertation. I should also mention that it was a great pleasure to work with all of them.