

Article

Low-Data Drug Design with Few-Shot Generative Domain Adaptation

Ke Liu ^{1,2}, Yuqiang Han ^{1,2,*}, Zhichen Gong ^{2,3} and Hongxia Xu ^{4,*}

¹ College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China; lk2017@zju.edu.cn

² ZJU-Hangzhou Global Scientific and Technological Innovation Center, Hangzhou 311200, China; zhichen.gong.18@ucl.ac.uk

³ Department of Computer Science, University College London, London WC1E 6BT, UK

⁴ Innovation Institute for Artificial Intelligence in Medicine, Zhejiang University, Hangzhou 310027, China

* Correspondence: hyq2015@zju.edu.cn (Y.H.); einstein@zju.edu.cn (H.X.)

Abstract: Developing new drugs for emerging diseases, such as COVID-19, is crucial for promoting public health. In recent years, the application of artificial intelligence (AI) has significantly advanced drug discovery pipelines. Generative models, such as generative adversarial networks (GANs), exhibit the potential for discovering novel drug molecules by relying on a vast number of training samples. However, for new diseases, only a few samples are typically available, posing a significant challenge to learning a generative model that produces both high-quality and diverse molecules under limited supervision. To address this low-data drug generation issue, we propose a novel molecule generative domain adaptation paradigm (Mol-GenDA), which transfers a pre-trained GAN on a large-scale drug molecule dataset to a new disease domain using only a few references. Specifically, we introduce a molecule adaptor into the GAN generator during the fine tuning, allowing the generator to reuse prior knowledge learned in pre-training to the greatest extent and maintain the quality and diversity of the generated molecules. Comprehensive downstream experiments demonstrate that Mol-GenDA can produce high-quality and diverse drug candidates. In summary, the proposed approach offers a promising solution to expedite drug discovery for new diseases, which could lead to the timely development of effective drugs to combat emerging outbreaks.

Keywords: drug design; domain adaptation; generative model



Citation: Liu, K.; Han, Y.; Gong, Z.; Xu, H. Low-Data Drug Design with Few-Shot Generative Domain Adaptation. *Bioengineering* **2023**, *10*, 1104. <https://doi.org/10.3390/bioengineering10091104>

Academic Editor: Antonio Brunetti

Received: 26 July 2023

Revised: 4 September 2023

Accepted: 18 September 2023

Published: 21 September 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Drug discovery and development are critical translational science activities that significantly contribute to human health and well-being [1]. However, drug discovery is a long-term, high-investment, and high-risk endeavor that traditionally relies on human expertise to design, synthesize, and test new drug molecules [2,3]. Traditional drug design methods can take an average of 6 to 12 years and cost billions of dollars to produce just one drug [4,5]. While only an estimated 10^8 compounds have ever been synthesized, the theoretical number of feasible compounds ranges from 10^{23} to 10^{60} [6]. As a result, conventional discovery methods can only explore a limited amount and diversity of chemical space. Therefore, there is an urgent need to develop efficient methods for exploring chemical space to accelerate and improve the drug discovery process.

In recent years, deep learning technology has been utilized to expedite and enhance the drug discovery process [7–9]. Specifically, bioinformatics scientists have shown a keen interest in generative models due to their remarkable capacity to comprehend and explore the intrinsic properties of data [10,11]. Rather than relying on human expertise to design molecules, generative models employ recent advancements in deep learning to tackle the inverse molecular design problem: determining the set of molecules that will satisfy a desired set of properties [3]. Generative models can swiftly identify a wide range of

molecules that are optimized for specific goals by mapping properties to structures. Recently, there has been a significant increase in the number and diversity of generative models employed in molecular design, such as variational autoencoders (VAEs) [12], generative adversarial networks (GANs) [13], and normalizing flow models [14]. In generative models, drug molecules are mostly represented as strings, such as SMILES (Simplified Molecular Input Line Entry System) [15] and SELFIES (Self-Referencing Embedded Strings) [16], or graphs [17]. For instance, VAEs have been utilized to generate SMILES strings and molecular graphs by approximately maximizing likelihood through variational inference techniques [18–20]. Similarly, GANs have also been adapted to generating molecules represented as sequences or graphs by formulating molecule generation as a minimax game [21–23]. Furthermore, normalizing flow models generate molecules by learning a series of invertible transformations between high-dimensional molecule data and a prior distribution [24–26].

Although generative models have made significant progress, their effectiveness primarily relies on the amount of training data, with larger sample sizes leading to greater accuracy. Unfortunately, acquiring labeled data for emerging diseases like COVID-19 can be challenging. The characteristics of effective drugs for such diseases are not yet established, and only a limited number of drugs are available to alleviate symptoms. As a result, training a generative model with adequate performance for low-data drug discovery is a daunting task due to the limited availability of labeled data. Few-shot generative domain adaptation has been introduced to address the challenge of limited data availability in GAN training [27–29]. Typically, a large-scale model is first trained in the source domain with a sufficient amount of data and then transferred to the target domain with only a few samples. Building on this idea, we propose a novel few-shot Molecule Generative Domain Adaptation paradigm (Mol-GenDA) for the low-data drug design. Specifically, we introduce a lightweight module called the molecule adaptor, which aids in adapting the generator to the target disease with the target molecule's attributes. We first pre-train the GAN on a large-scale drug-like dataset, then freeze the parameters of the pre-trained generator and optimize only the molecule adaptor during fine tuning on the new disease dataset. This approach leverages the prior knowledge learned in the source domain to inherit the generation quality and diversity of the source model.

We have conducted extensive experiments to evaluate the proposed method's ability to generate molecules with specific structures and desired properties in low-data drug design. The experiments showed that the proposed method can generate both simple structural features, such as halogen groups or aromatic rings, and more complex molecules with higher scores of desired properties, such as penalized logP and quantitative estimate of drug-likeness (QED score), among others. As part of our study, we have designed drugs that could be effective against COVID-19 and assessed the properties of various drug candidates.

2. Research Problem and Motivation

2.1. Research Problem

The aim of this study is to develop a generation method for the rapid design of effective drugs for emergent diseases. The main challenge in achieving this goal is training an effective generative model on only a few referenced drug molecules. Moreover, emerging diseases often require drugs with multiple desired properties, further complicating the generation process. Therefore, the generation method needs to address these challenges, namely few-shot reference drugs and multiple desired properties. The low-data drug design problem can be formalized as follows: If we only have a few known drug molecules \mathcal{M}_r that are partially effective in treating a specific disease, such as relieving certain symptoms, how can we train a generative model f_θ to design new drugs based on this information?

Generative models are expected to possess two key capabilities: (1) structure-constrained generation, which involves the ability to make simple structural modifications such as altering the presence of halogen groups or adjusting the number of aromatic rings; and

(2) property-constrained generation, which enables the model to generate molecules with higher scores of desired properties. By leveraging these abilities, generative models can produce drug molecules that are more effective in treating emerging diseases or are easier to manufacture.

2.2. Limitation of Previous Methods

VAEs. Variational autoencoders (VAEs) are widely used in drug design and consist of an encoder and a decoder [12]. The encoder converts a molecule into a latent vector representation and maps it to a pre-defined distribution of valid molecule latent vectors. Novel molecules can be generated by sampling latent vectors from the distribution and decoding them with the decoder [20,30]. For instance, the JT-VAE method interpolates reference drugs within the pre-trained VAE's latent space [20], while GF-VAE randomly samples the space surrounding reference drug molecules in the latent space [31]. However, these approaches have struggled to produce molecules that exhibit both diversity and desired properties.

GANs. Generative adversarial networks (GANs) have become widely used in various fields, including image, audio, and video processing [32]. A GAN comprises two parts: a generator G and a discriminator D . During training, the GAN plays a max-min game, in which D learns to differentiate between real and generated data from G , while G learns to generate more realistic data to deceive D . Ultimately, the trained G generates realistic data, and the trained D improves its ability to classify fake data. In recent years, there has been a surge of GAN-based models applied to molecule design [23,33–35]. For example, Mol-GAN trains a GAN from scratch using a large dataset of drug molecules with desired properties [23]. Mol-CycleGAN, on the other hand, trains GANs based on the latent space of pre-trained VAEs [35]. However, these methods require extensive collections of drug molecules for training.

To summarize, none of the previous methods have addressed the challenge of generating a diverse set of desired drugs with only a few references. Few-shot generation has been extensively studied in computer vision [36–38]; however, few-shot drug molecule generation remains an ongoing area [39]. Low-data drug discovery techniques, such as few-shot property prediction [40], cannot be directly applied to molecule generation.

3. Our Method

To tackle the issue of low-data drug molecule generation, we propose a novel generative domain adaptation approach called Mol-GenDA, inspired by recent work in computer vision [38]. Figure 1 depicts the overall workflow of Mol-GenDA. Firstly, the GAN is pre-trained on a large-scale drug molecule dataset. Then, it is fine-tuned with few-shot reference drug molecules using a lightweight molecule adaptor. Finally, the model is used to generate desired drug molecules.

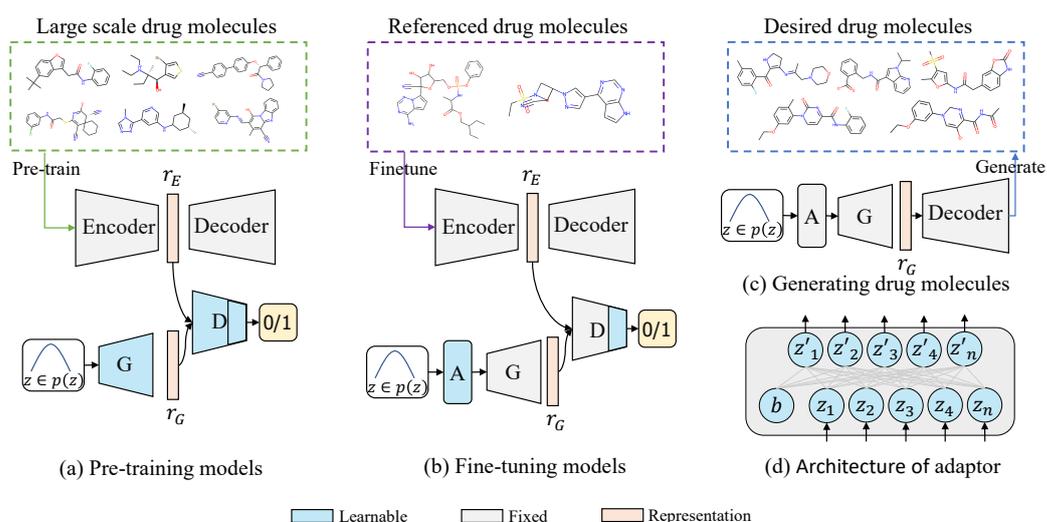


Figure 1. An illustrative diagram of Mol-GenDA. (a) The GAN is pre-trained on a large-scale drug molecule dataset. (b) For a specific task, the pre-trained GAN is fine-tuned with related drug molecules using a molecule adaptor. (c) The fine-tuned GAN is utilized to generate desired drug molecules. (d) The architecture of molecule adaptor, where a two-layer neural network is adopted to adapt the original distribution to that of desired drug molecules. A, G, and D denote the molecule adaptor, generator, and discriminator, respectively. Best viewed in color.

3.1. Large-Scale Pre-Training

We adopted the Junction Tree Variational Autoencoder (JT-VAE) to encode drug molecules into a latent space and decode latent vectors back to drug molecules, as in previous studies on drug molecule generation [23,35,41]. The JT-VAE approach is VAE-based and operates on the graph structure representation of molecules, employing a junction-tree scaffold of molecule sub-components and a graph-structure representation of molecules. Compared to other VAE-based methods that operate on the SMILES representation of molecules, JT-VAE exhibits superior performance, with 100% validity of the generated molecules [30,31]. Pre-training the JT-VAE on a large-scale molecule dataset can improve its representation learning capability. In this work, we take the pre-trained JT-VAE (<https://github.com/wengong-jin/icml18-jtnn>, accessed on 25 April 2021) on ZINC-250K from previous work [20]. During pre-training and fine tuning, we froze the parameters of JT-VAE.

In this stage, we pre-train the GAN on a large-scale molecule dataset and freeze the JT-VAE. As shown in Figure 1a, the generator G produces fake molecule latent vectors, while the VAE encoder produces realistic ones. The discriminator D is trained to classify whether the vectors are realistic or generated by G . The training strategy is a max-min game, and the objective between the generator G and discriminator D can be formulated as:

$$\min_G \max_D \mathbb{E}_{r_E \sim \mathbb{P}_r} [\log(D(r_E))] + \mathbb{E}_{r_G \sim \mathbb{P}_g} [\log(1 - D(r_G))], \quad (1)$$

where \mathbb{P}_r is the data distribution, \mathbb{P}_g is the model distribution defined implicitly by $r_G = G(z)$, and $z \sim p(z)$ is sampled from a simple noise distribution (a Gaussian distribution is chosen in this work). Following WGAN-GP [42], we introduce the gradient penalty to the model, and the loss function is finally improved as follows:

$$L = \mathbb{E}_{r_G \sim \mathbb{P}_g} [D(r_G)] - \mathbb{E}_{r_E \sim \mathbb{P}_r} [D(r_E)] + \lambda \mathbb{E}_{\hat{r}_E \sim \mathbb{P}_{\hat{r}_E}} \left[\left(\|\nabla_{\hat{r}_E} D(\hat{r}_E)\|_2 - 1 \right)^2 \right], \quad (2)$$

where $\hat{r}_E \sim \mathbb{P}_{\hat{r}_E}$ is uniformly sampled along the straight lines between pairs of points sampled from the data distribution \mathbb{P}_r and the generator distribution \mathbb{P}_g . Algorithm 1 summarizes the details of the pre-training process.

Algorithm 1: Large-scale pre-training

Input: \mathcal{M}_I , a large-scale drug molecules dataset;
 θ_E , pre-trained encoder parameters;
 θ_D , pre-trained decoder parameters;
 θ_G , initial generator parameters;
 θ_C , initial discriminator parameters;
 z , random noise.

Output: $\hat{\theta}_G$, pre-trained generator parameters;
 θ'_C , pre-trained discriminator parameters.

Hyperparameters: $N_{pre} \in \mathbb{Z}^*$.

```

1 for  $i = 1, 2, \dots, N_{pre}$  do
2    $r_E \leftarrow \theta_E(M_I)$ 
3    $r_G \leftarrow \theta_G(z)$ 
4    $\text{loss}(\theta_G, \theta_C) = L_\theta(r_E, r_G)$  based on Equation (2)
5    $(\theta_G, \theta_C) \leftarrow (\theta_G, \theta_C) - \eta \cdot \nabla \text{loss}(\theta_G, \theta_C)$ 
6 end
7  $\hat{\theta}_G = \theta_G, \theta'_C = \theta_C$ 
8 return  $\hat{\theta}_G, \theta'_C$ 

```

3.2. Generative Domain Adaptation

The pre-trained generator of Mol-GenDA has already acquired the ability to generate diverse and high-quality drug molecules, learned from a large-scale drug molecule dataset. The random noise z input into the generator of GAN can be viewed as the features used to generate latent drug molecule vectors in the latent space of a pre-trained JT-VAE [43]. These latent vectors can then be decoded by the decoder of JT-VAE to produce drug molecules. The goal of Mol-GenDA fine tuning is to learn the ability to select appropriate features, i.e., the noise z , for generating desired drug molecules. However, fine-tuning all the parameters of the generator on a few reference drug molecules carries the risk of overfitting. To address these concerns, we designed a lightweight module called molecule adaptor. As illustrated in Figure 1b, during the fine-tuning process, we freeze the parameters of the generator and only update those of the molecule adaptor. The goal is to transfer the noise distribution, which serves as the input features of the generator to produce latent vectors, to the distribution of reference drug molecules. The architecture of the molecule adaptor is illustrated in Figure 1d, and it can be defined as:

$$z' = Az + b \quad (3)$$

where A and b are the linear projection matrix that controls the variation scale of the latent vector, and a bias vector in the affine module learned in the fine-tuning.

Regarding the discriminator, the first several layers are responsible for feature extraction, while the latter layers perform classification [44]. As the discriminator is a binary classifier, we freeze the first several layers and train only the last n layers, aiming to maintain the discriminator's ability to extract key features while training it to classify whether the drug is desired or not. In this work, we update the last two layers of the discriminator during fine-tuning. Reference drug molecules are fed into the joint model, and the objective is the same as in pre-training, as shown in Equation (2). Algorithm 2 summarizes the details of the generative domain adaptation process.

Algorithm 2: Generative Domain Adaptation Fine Tuning

Input: \mathcal{M}_r , a dataset of referenced drug molecules;
 θ_E , pre-trained encoder parameters;
 θ_D , pre-trained decoder parameters;
 $\hat{\theta}_G$, pre-trained generator parameters;
 θ'_C , pre-trained discriminator parameters;
 θ_A , initial adaptor parameters;
 z , random noise.

Output: $\hat{\theta}_C$, fine-tuned discriminator parameters;
 $\hat{\theta}_A$, fine-tuned adaptor parameters.

Hyperparameters: $N_{adapt} \in \mathbb{Z}^*$.

```

1  $\theta''_C \leftarrow \text{Last2}(\theta'_C)$  ▷ get the last two layers of discriminator
2 for  $i = 1, 2, \dots, N_{adapt}$  do
3    $r_E \leftarrow \theta_E(\mathcal{M}_r)$ 
4    $r_G \leftarrow \theta_G(\theta_A(z))$ 
5    $\text{loss}(\theta_A, \theta'_C) = L''_{\theta}(r_E, r_G)$  based on Equation (2)
6    $\theta_A, \theta''_C \leftarrow (\theta_A, \theta''_C) - \eta \cdot \nabla \text{loss}(\theta_A, \theta'_C)$ 
7    $\text{Last2}(\theta''_C) \leftarrow \theta''_C$ 
8 end
9  $\hat{\theta}_A = \theta_A, \hat{\theta}_C = \theta''_C$  return  $\hat{\theta}_A, \hat{\theta}_C$ 

```

3.3. Constrained Molecule Generation

Structure- and property-constrained molecule generation are two common tasks in drug design, and Mol-GenDA can solve their low-data problem.

Structure-constrained generation. To achieve structure-constrained generation, we fine-tune the molecule adaptor $\hat{\theta}_A$ on few-shot molecules with specific structures. With the additional pre-trained generator $\hat{\theta}_G$ in GAN and decoder $\hat{\theta}_D$ in VAE, the model generates molecules with desired structures as follows:

$$\mathcal{M}_{gen} = \hat{\theta}_D(\hat{\theta}_G(\hat{\theta}_A(z))), \quad (4)$$

where z is randomly sampled noise. Specifically, to generate desired drug molecules, the molecule adaptor adapts the noise distribution to the desired molecule distribution. Then, the noise is input into the generator to obtain the latent vectors r_G , which can be decoded to the desired drug molecules \mathcal{M}_{gen} using the decoder.

Property-constrained generation. Generating molecules with desired properties follow a similar process to structure-constrained generation, with the only difference being that the molecule adaptor is fine-tuned on references with high scores for specific properties. Algorithm 3 summarizes the details of constrained molecule generation.

Algorithm 3: Constrained Molecule Generation

Input: $\hat{\theta}_A$, trained adaptor parameters;
 $\hat{\theta}_G$, pre-trained generator parameters;
 θ_D , pre-trained decoder parameters;
 z , random noise.

Output: \mathcal{M}_{gen} , generated desired drug molecules;

Hyperparameters: $N_{gen} \in \mathbb{Z}^*$

```

1  $r_G \leftarrow \hat{\theta}_G(\hat{\theta}_A(z))$ 
2  $\mathcal{M}_{gen} \leftarrow \hat{\theta}_D(r_G)$ 
3 return  $\mathcal{M}_{gen}$ 

```

4. Results

4.1. Data

The model was pre-trained on the ZINC-250K dataset, which contains 250,000 drug-like molecules extracted from the ZINC database [45]. This dataset is commonly used in similar studies [20,35]. To evaluate the model's performance on few-shot molecule generation, we tested it on six datasets, including the structure- and property-constrained generation. The statistics of these datasets are presented in Table 1.

Table 1. Statistics of the datasets, where Num. and Diver. indicate number and diversity, respectively.

Task	Dataset	Num.	Diver.	Plogp	QED	1 Ring	2 Rings	3 Rings	Halogen	
Pre-training	ZINC	2.5×10^5	0.915	0.561	0.731	75,580	98,222	47,603	87,556	
Structure-constrained	1 ring	5	0.902	0.519	0.799	5	-	-	-	
		10	0.905	-	-	10	-	-	-	
		50	0.906	-	-	50	-	-	-	
		100	0.909	-	-	100	-	-	-	
	2 rings	5	0.893	0.668	0.839	-	5	-	-	
		10	0.905	-	-	-	10	-	-	
		50	0.906	-	-	-	50	-	-	
		100	0.908	-	-	-	100	-	-	
	3 rings	5	0.889	-	-	-	-	5	-	
		10	0.902	-	-	-	-	10	-	
		50	0.907	-	-	-	-	50	-	
		100	0.908	-	-	-	-	100	-	
	Halogen	5	0.895	-	-	-	1	-	3	5
		10	0.900	-	-	-	2	3	3	10
		50	0.908	-	-	-	11	28	6	50
		100	0.911	-	-	-	25	52	15	100
Property-constrained	QED	5	0.864	0.604	0.947	-	-	-	-	
		10	0.888	0.599	0.947	-	-	-	-	
		50	0.891	0.602	0.947	-	-	-	-	
		100	0.893	0.602	0.947	-	-	-	-	
	PlogP	5	0.878	1.000	0.292	-	-	-	-	
		10	0.891	1.000	0.288	-	-	-	-	
		50	0.901	1.000	0.357	-	-	-	-	
		100	0.905	1.000	0.390	-	-	-	-	

Structure-constrained generation. For downstream tasks of structure-constrained generation, we constructed four subsets with specific structures. Aromatic rings and halogen moieties are two important structural properties of molecules. The **1 ring**, **2 rings**, and **3 rings** datasets contain molecules with one, two, and three rings, respectively. The **Halogen** dataset contains molecules with halogen groups. Each of these datasets contains four subsets of 5-shot, 10-shot, 50-shot, and 100-shot reference drug molecules. To construct these datasets, we randomly selected 100 molecules from the ZINC-250K dataset for each 100-shot dataset. Then, we selected 50 molecules from the 100-shot subsets to construct the 50-shot subsets. The 10-shot subsets were randomly selected from the 50-shot subsets, and the 5-shot subsets were selected from the 10-shot ones. They were constructed to evaluate the model's ability to perform structural transformations. For instance, the model can execute simple structural modifications such as changing the presence of halogen groups or altering the number of aromatic rings.

Property-constrained generation. **Plogp** and **QED** datasets consist of molecules with top penalized logP (Plogp) and quantitative estimate of drug-likeness (QED) scores, respectively. These datasets evaluate the performance of property-constrained generation with few-shot references. Specifically, we extracted the molecules with the first 5, 10, 50, and 100

top scores of QED and Plogp from the ZINC-250K dataset to create the 5-shot, 10-shot, 50-shot, and 100-shot datasets.

4.2. Model and Training Configurations

In our evaluation, the generator G of GAN consists of a seven-layer multilayer perceptron (MLP) with 100, 128, 256, 256, 512, 256, and 56 neurons, respectively. The discriminator D of GAN contains a 5-layer MLP with 56, 128, 256, 128, and 1 neurons, respectively. The activation functions in the generator and discriminator are $Tanh()$ and $LeakyReLU()$, respectively. The adaptor is composed of two-layer MLPs with 57 and 56 neurons, respectively. The architecture of an adaptor is straightforwardly designed according to the length of the latent vector. We pre-trained the GANs for 200 epochs with a mini-batch size of 128 and optimized the objective using the Adam optimizer [46] with a learning rate of $1e-3$. The loss was calculated using Equation (2). During the fine-tuning process, we also used the Adam optimizer and fine-tuned the model for 40 epochs with a mini-batch size of 1. For each experiment, we generated 1,000 drug molecules for evaluation. All experiments were conducted on a computing cluster with eight NVIDIA® GeForce® RTX 2080 Ti 11GB GPUs and an Intel® Xeon® Gold 6139 CPU @ 2.30GHz. PyTorch [47] was applied to complete our model and RDKit [48] was used to draw the pictures and estimate the properties of molecules.

4.3. Comparison to Previous Methods

Structure-constrained generation. We compare our proposed method with GAN and pre-trained GAN, for structure-constrained generation. GAN is directly trained with few-shot drug molecule references with specific structures in the latent space of VAE. In this work, we adopted WGAN-GP [42]. On the other hand, **pre-trained GAN** is pre-trained on the ZINC-250K dataset from scratch.

Property-constrained generation. In addition to GAN and pre-trained GAN, we compare our proposed method with previous approaches that use reference drugs, including interpolating [31] and random sampling [30] for property-constrained generation. In **random sampling**, the reference drugs are encoded into the latent space using the VAE encoder to obtain their representations. Then, the spaces around these points are randomly sampled with radii of 0.5, 1, and 2, respectively, as used in this work. On the other hand, in **interpolation**, the desired drugs in the latent space are obtained by interpolating between each pair of reference drug latent vectors.

4.4. Evaluation Metrics

Structure-constrained generation. We evaluate the performance of structure-constrained generation based on diversity, uniqueness, and quality. **Diversity** measures the diversity of generated molecules and is defined as:

$$\text{Diversity} = 1 - \frac{1}{|M_{gen}|(|M_{gen}| - 1)} \sum_{m_1, m_2 \in M_{gen}}^{m_1 \neq m_2} sim(m_1, m_2), \quad (5)$$

where $|\cdot|$, $sim(\cdot)$, and M_{gen} denote the operation of obtaining the number, a similarity calculation method, and the generated molecules, respectively. In this work, we adopt the Tanimoto similarity between two extended-connectivity fingerprint bit vectors. **Uniqueness** measures the degree of variety during sampling and is defined as the ratio between the number of unique samples and valid samples:

$$\text{Uniqueness} = \frac{|M_{unique}|}{|M_{gen}|}, \quad (6)$$

where M_{unique} denotes the set of unique drugs (i.e., removing duplicated drugs in the generated set). **Quality** is the ratio between the numbers of drug molecules with desired structures and the generated drug molecules, defined as follows:

$$\text{Quality} = \frac{|M_{desired}|}{|M_{gen}|}, \quad (7)$$

where $M_{desired}$ is the desired drug molecule sets without duplicated molecules.

Property-constrained generation. In addition to diversity and uniqueness, we evaluate the performance of property-constrained generation based on the scores of desired properties, including penalized logP (**PlogP**) and quantitative estimate of drug-likeness (**QED**). PlogP is a commonly used property to evaluate molecule optimization models' performance, as it is relevant in the drug design process. It is defined as the logarithm of the ratio of the concentrations of a solute in two solvents, and it provides a measure of lipophilicity. QED score is another critical metric for drug design, which measures the similarity between a compound's properties and those of known drugs. QED stands for quantitative estimate of drug-likeness, and it is a widely used measure in drug discovery.

4.5. Performance and Discussion

4.5.1. Structure-Constrained Generation

The results of generating molecules with one, two, and three aromatic rings, as well as halogens, are presented in Figure 2. Overall, Mol-GenDA demonstrates superior performance compared to GAN and pre-trained GAN in terms of diversity, quality, and uniqueness in most cases. While Mol-GenDA performs slightly worse in terms of the uniqueness of generated molecules with three rings, the experimental results still demonstrate its ability to enhance the quality of generation while maintaining diversity.

Specifically, GAN trained from scratch on few-shot reference drug molecules performs worse than both Mol-GenDA and pre-trained GAN, as training a GAN with just a few molecules is challenging. Although the quality and diversity of molecules generated by GAN increase with more reference molecules for training, it is still not enough to train a proper GAN with just 100 reference drug molecules. Pre-trained GAN generates more diversified drug molecules than GAN because GAN is trained with only a few reference drug molecules, which limits its learning space. Additionally, the training data for GAN are only a subset of those of pre-trained GAN, further narrowing down its learning space.

Mol-GenDA outperforms GAN in terms of diversity because the pre-trained generator maintains the knowledge learned from large-scale training drug molecules in pre-training. The diversity of drug molecules generated by pre-trained GAN and Mol-GenDA is similar because both learned from large-scale training drug molecules. Moreover, the diversity of molecules generated by Mol-GenDA varies in a small range since the diversity learned from large-scale pre-training is maintained by freezing the parameters of the generator in the domain adaptation fine tuning.

Mol-GenDA generates more desired drug molecules than pre-trained GAN because of the additional fine-tuning stage that adapts the pre-trained GAN to a specific domain. Training a GAN from scratch is hard because the latent space for the GAN to explore is too large while only a few reference drug molecules are available, making it difficult to learn the common features of the reference drug molecules. In contrast, only part of Mol-GenDA's parameters are updated in fine tuning, allowing for generating drug molecules with higher quality than GAN and pre-trained GAN.

Figure 3 shows the generation examples of Mol-GenDA, with molecules randomly selected from the generation of each task. We can see that Mol-GenDA successfully generates molecules with desired structures.

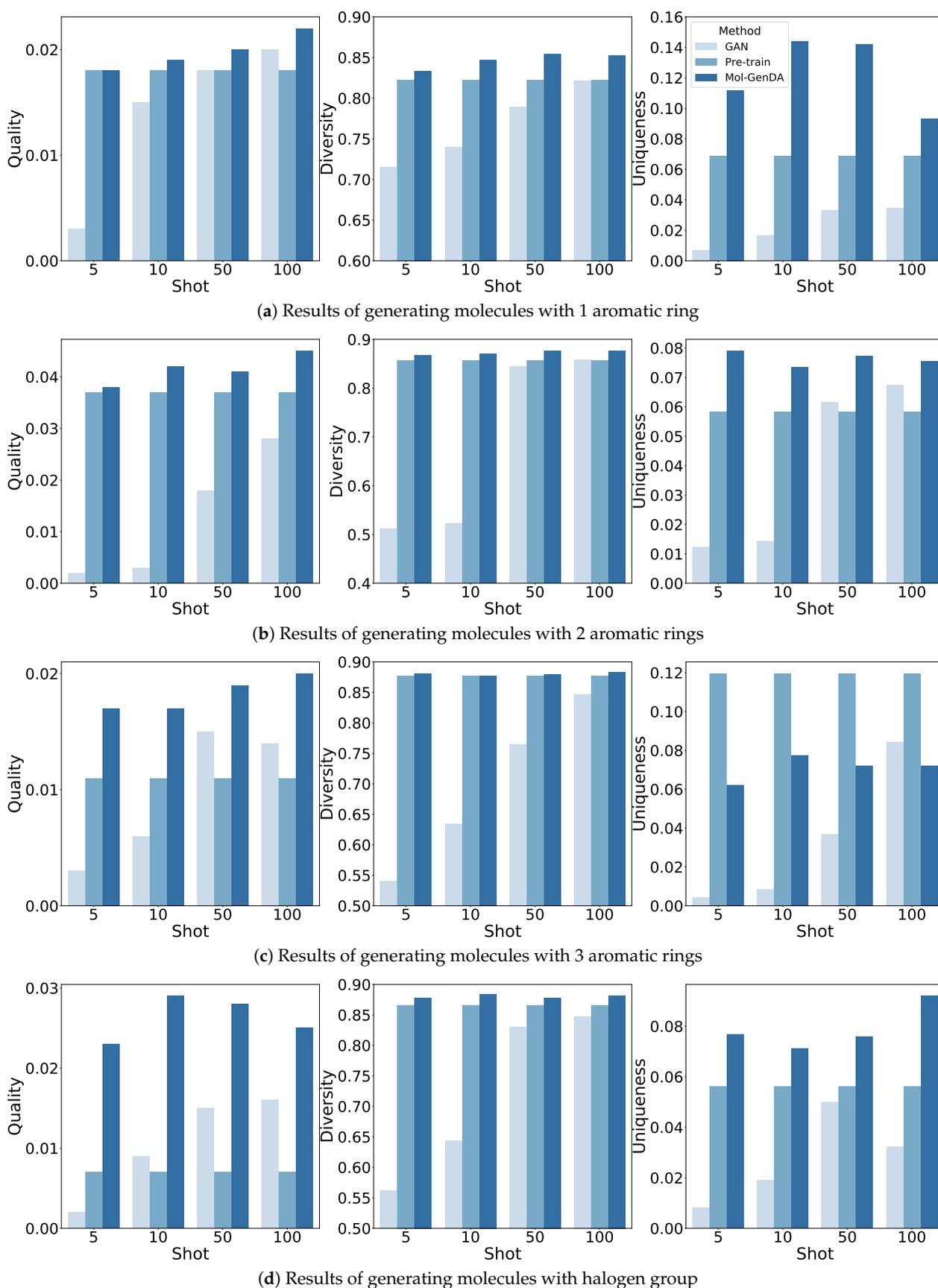


Figure 2. Results of structure-constrained molecule generation.

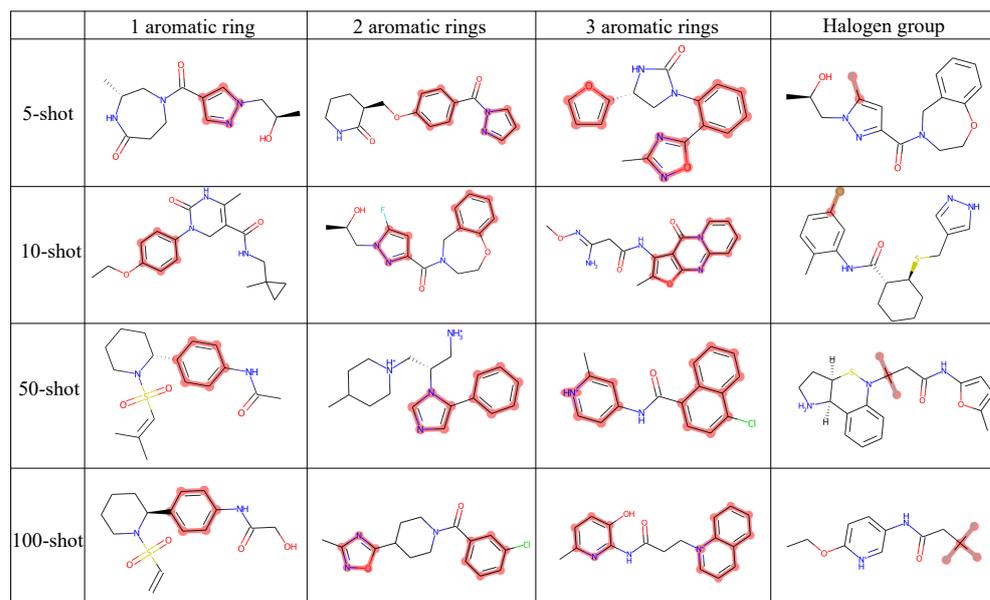


Figure 3. Structure-constrained generation examples of Mol-GenDA. Each row represents the structures that are desired and used for Mol-GenDA training, including 1, 2, and 3 aromatic rings, and the halogen group. Each column corresponds to the generation with 5-shot, 10-shot, 50-shot, and 100-shot reference drug molecules, respectively. The red highlight indicates the corresponding structures, i.e., the aromatic ring and halogen group. All molecules shown in the figure are randomly selected from the corresponding task's generation.

4.5.2. Property-Constrained Generation

The QED- and PlogP-constrained drug molecule generation results are presented in Tables 2 and 3, respectively. Mol-GenDA outperforms other methods in terms of QED and PlogP scores, demonstrating its capability for few-shot property-constrained molecule generation.

Table 2. Results of QED-constrained generation.

		Inter-Polate	Random Sampling			GAN	Pre-Train GAN	Mol-GenDA
			0.5	1	2			
5-shot	QED	0.681	0.486	0.519	0.497	0.729	0.749	0.771
	Diversity	0.868	0.908	0.923	0.930	0.850	0.866	0.859
	Uniqueness	0.129	0.929	0.979	0.994	0.094	0.243	0.240
10-shot	QED	0.738	0.439	0.525	0.516	0.753	0.749	0.769
	Diversity	0.886	0.902	0.920	0.927	0.832	0.866	0.865
	Uniqueness	0.395	0.929	0.979	0.994	0.900	0.243	0.246
50-shot	QED	0.721	0.439	0.506	0.501	0.736	0.749	0.749
	Diversity	0.897	0.910	0.922	0.928	0.841	0.866	0.862
	Uniqueness	0.998	0.840	0.997	0.996	0.073	0.242	0.243
100-shot	QED	0.722	0.467	0.521	0.683	0.748	0.749	0.762
	Diversity	0.897	0.914	0.921	0.927	0.852	0.866	0.866
	Uniqueness	0.988	0.897	0.999	0.997	0.066	0.243	0.263

Table 3. Results of PlogP-constrained generation.

		Inter-Polate	Random Sampling			GAN	Pre-Train GAN	Mol-GenDA
			0.5	1	2			
5-shot	Plogp	0.651	0.639	0.547	0.519	0.519	0.568	0.682
	Diversity	0.880	0.908	0.923	0.930	0.825	0.866	0.936
	Uniqueness	0.234	0.929	0.979	0.994	0.069	0.243	0.252
10-shot	Plogp	0.660	0.669	0.667	0.609	0.562	0.568	0.679
	Diversity	0.899	0.871	0.920	0.927	0.837	0.866	0.866
	Uniqueness	0.612	0.827	0.994	0.995	0.065	0.243	0.255
50-shot	Plogp	0.635	0.617	0.614	0.594	0.547	0.568	0.677
	Diversity	0.893	0.910	0.922	0.928	0.849	0.866	0.865
	Uniqueness	0.982	0.840	0.997	0.996	0.068	0.243	0.232
100-shot	Plogp	0.659	0.653	0.607	0.590	0.484	0.568	0.672
	Diversity	0.913	0.914	0.921	0.928	0.864	0.866	0.865
	Uniqueness	0.985	0.897	0.999	0.994	0.085	0.243	0.236

Interpolation and random sampling methods generate drug molecules with similar representation vectors in the latent space, and increasing the radius of random sampling improves the diversity of generated molecules by ensuring more diverse representation vectors. However, learning the target property features from few-shot reference drug molecules is challenging. Pre-trained GANs are trained on large-scale drug molecules without accessing the specific property features of the target molecules, which limits their ability to generate desired molecules. Similarly, GANs trained from scratch also struggle to capture the target features necessary for generating the desired molecules, although their performance can be improved with the aid of a well-trained VAE.

Mol-GenDA outperforms other baseline methods in terms of property score, but it performs worse in diversity than interpolation and random sampling, and better than GANs and pre-trained GANs. Since the representation vectors generated by interpolation and random sampling are different naturally, the diversity of the molecules decoded from these vectors is guaranteed, but they fail to generate molecule drugs with desired properties. Overall, Mol-GenDA appropriately addresses the challenge of generating desired drug molecules with few-shot references.

Figure 4 displays the generated candidate molecules with the highest property scores for their respective tasks. We can see that Mol-GenDA successfully generates molecules with higher scores of desired properties.

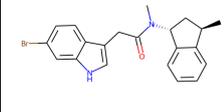
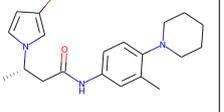
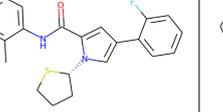
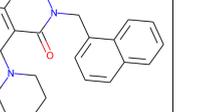
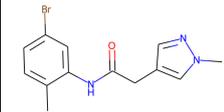
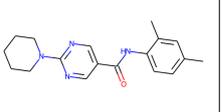
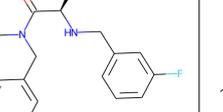
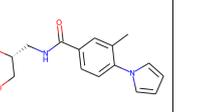
	5-shot	10-shot	50-shot	100-shot
PlogP	 0.894	 0.889	 0.980	 0.964
QED	 0.947	 0.944	 0.941	 0.944

Figure 4. Property-constrained generation examples of Mol-GenDA. Each row represents the generation using 5-shot, 10-shot, 50-shot, and 100-shot reference drug molecules, respectively. Each column indicates the desired properties used for Mol-GenDA training, including QED and PlogP. The selected molecules in each row have the highest property scores for their corresponding task. The property scores are below each molecule.

4.5.3. Activity: Dopamine Receptor D2

To further validate the effectiveness of our Mol-GenDA, we conduct experiments on generating bioactive drug molecules with Dopamine Receptor D2 (DRD2) as the biological target. DRD2 score is the probability that one molecule can trigger the biological activity of DRD2. Specifically, we utilize a machine learning-based score model from [35] to select the top five reference molecules that can trigger the biological activity of DRD2. Mol-GenDA then generates desired molecules with the five reference molecules. Figure 5 shows the top five molecules generated by Mol-GenDA and GAN trained from scratch in terms of DRD2 score. We can see that the molecules generated by Mol-GenDA share similar substructures with reference molecules and maintain the diversity score of 0.921, which is higher than that of GAN (i.e., 0.770). Furthermore, Mol-GenDA generates drug molecules with much higher DRD2 scores (i.e., the average DRD2 score of the top five molecules is 0.544) than those of GAN (i.e., the average DRD2 score is 0.048). Mol-GenDA outperforms GAN trained from scratch on both DRD2 score and diversity because of the fine-tuning paradigm in Mol-GenDA, which generates molecules with desired properties and maintains diversity. The pre-training enables Mol-GenDA to generate valid molecules, and the fine-tuning paradigm introduces an adaptor to learn the distribution of drug molecules with desired properties while freezing the parameters of the generator, which maintains the diversity of generated drug molecules. Additionally, during the fine-tuning process, only the last two layers of the discriminator are trained, which simplifies the training procedure. However, for GAN trained from scratch, exploring the enormous latent space for drug molecules with desired properties is difficult. Furthermore, we find that the QED and PlogP scores of Mol-GenDA (i.e., 0.724 and 0.491) are comparable to those of GAN (i.e., 0.756 and 0.503). Similarly, the two methods also achieved comparable performance on the other two properties, drug candidate score (DCS) (i.e., 0.595 and 0.609) and synthetic accessibility (SA) (i.e., 0.531 and 0.503). This is because both of them contain the VAE module which is taken from previous works [20] and pre-trained to be full of valid drug molecules in the latent space for both methods.

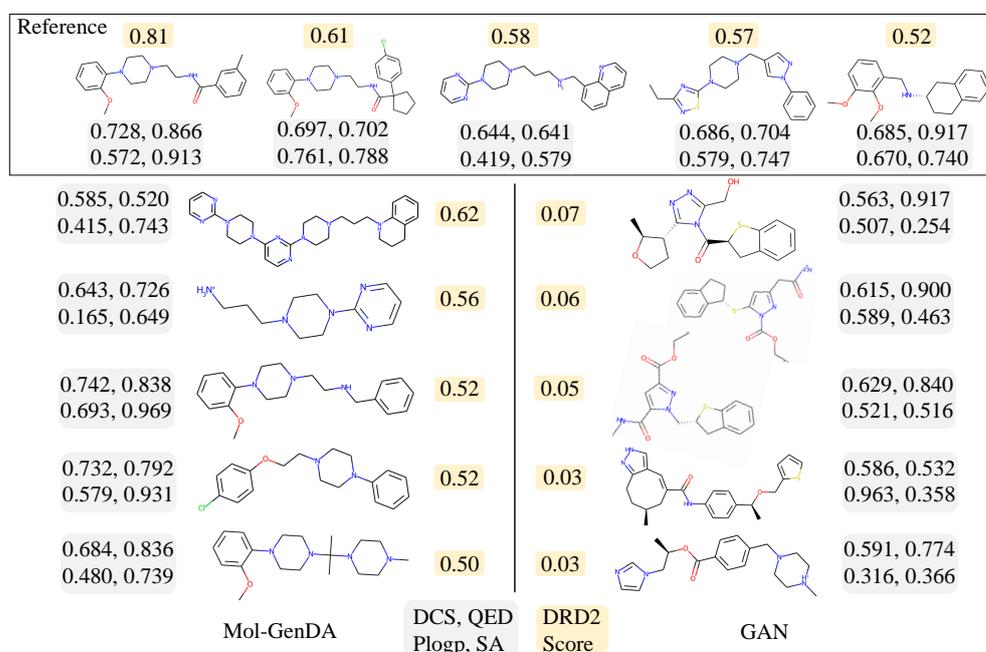


Figure 5. The 5 reference drugs and candidates generated by Mol-GenDA and GAN for DRD2. The numbers beside each molecule represent the drug candidate scores (DCS), QED, PlogP, SA, respectively, with higher scores indicating better generation results. DRD2 is the probability that one molecule can trigger the biological activity of DRD2. The drug molecules are arranged in order of DRD2 Score.

Extensive experimental results on both structure-constrained and property-constrained generation consistently demonstrated the superiority of Mol-GenDA over previous works in terms of common evaluation metrics on the low-data drug design task.

Limitations. In the few-shot generation setting, Mol-GenDA outperformed previous works in structure-constrained generation and most methods in property-constrained generation. However, its diversity in generating molecules with desired properties is not as good as that of interpolation and random sampling methods. This is due to the limited space explored by Mol-GenDA during fine-tuning, as only the adaptor and the last two layers of the discriminator are updated, resulting in a trade-off between diversity and desired properties.

Although Mol-GenDA still requires a few reference drugs, our future work will focus on generating desired drugs with one-shot learning or directly controlling the structures of generated drugs. Specifically, we aim to generate desired drug molecules using a large chemical language model, utilizing only the description of the molecule properties or structures to retrieve the knowledge from the large chemical language model.

Author Contributions: Conceptualization, K.L.; methodology, K.L.; software, K.L.; validation, K.L.; formal analysis, K.L. and Y.H.; investigation, K.L.; resources, K.L., Y.H. and H.X.; data curation, K.L.; writing—original draft preparation, K.L.; writing—review and editing, K.L., Y.H., Z.G. and H.X.; visualization, K.L.; supervision, Y.H.; project administration, Y.H. and H.X.; funding acquisition, H.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China under grants No. 82202984.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets employed in our study can be found at Google Drive: <https://drive.google.com/drive/folders/10Z2n6co40abDIkuNlrNgDZkO1e7K-g91?usp=sharing>, accessed on 11 January 2022. All code has been deposited and is publicly available on Github: <https://github.com/zjuKeLiu/Mol-GenDA>, accessed on 3 May 2023.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Salazar, D.E.; Gormley, G. Chapter 41—Modern Drug Discovery and Development. In *Clinical and Translational Science*, 2nd ed.; Robertson, D., Williams, G.H., Eds.; Academic Press: Cambridge, MA, USA, 2017; pp. 719–743. [CrossRef]
2. Dowden, H.; Munro, J. Trends in clinical success rates and therapeutic focus. *Nat. Rev. Drug Discov.* **2019**, *18*, 495–496. [CrossRef] [PubMed]
3. Bilodeau, C.; Jin, W.; Jaakkola, T.; Barzilay, R.; Jensen, K.F. Generative models for molecular discovery: Recent advances and challenges. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2022**, *12*, e1608. [CrossRef]
4. Sharma, N.; Bora, K.S. Computer Aided Drug Design, 3D Printing, and Virtual Screening: Recent Advancement and Applications in the Pharma Field. *ECS Trans.* **2022**, *107*, 16423–16430. [CrossRef]
5. Mullard, A. New drugs cost US \$2.6 billion to develop. *Nat. Rev. Drug Discov.* **2014**, *13*, 877. [CrossRef]
6. Reymond, J.L.; Awale, M. Exploring chemical space for drug discovery using the chemical universe database. *ACS Chem. Neurosci.* **2012**, *3*, 649–657. [CrossRef]
7. Schneider, P.; Walters, W.P.; Plowright, A.T.; Sieroka, N.; Listgarten, J.; Goodnow, R.A., Jr.; Fisher, J.; Jansen, J.M.; Duca, J.S.; Rush, T.S.; et al. Rethinking drug design in the artificial intelligence era. *Nat. Rev. Drug Discov.* **2020**, *19*, 353–364. [CrossRef]
8. Jing, X.; Xu, J. Fast and effective protein model refinement using deep graph neural networks. *Nat. Comput. Sci.* **2021**, *1*, 462–469. [CrossRef]
9. Chen, Z.; Min, M.R.; Parthasarathy, S.; Ning, X. A deep generative model for molecule optimization via one fragment modification. *Nat. Mach. Intell.* **2021**, *3*, 1040–1049. [CrossRef]
10. Butler, K.T.; Davies, D.W.; Cartwright, H.; Isayev, O.; Walsh, A. Machine learning for molecular and materials science. *Nature* **2018**, *559*, 547–555. [CrossRef]
11. Walters, W.P.; Barzilay, R. Applications of deep learning in molecule generation and molecular property prediction. *Accounts Chem. Res.* **2020**, *54*, 263–270. [CrossRef]
12. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. In Proceedings of the 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, 14–16 April 2014.

13. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 2672–2680.
14. Rezende, D.; Mohamed, S. Variational inference with normalizing flows. In Proceedings of the International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 1530–1538.
15. Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36. [[CrossRef](#)]
16. Krenn, M.; Häse, F.; Nigam, A.; Friederich, P.; Aspuru-Guzik, A. Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. *Mach. Learn. Sci. Technol.* **2020**, *1*, 045024. [[CrossRef](#)]
17. Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; Riley, P. Molecular graph convolutions: Moving beyond fingerprints. *J. Comput.-Aided Mol. Des.* **2016**, *30*, 595–608. [[CrossRef](#)] [[PubMed](#)]
18. Kusner, M.J.; Paige, B.; Hernández-Lobato, J.M. Grammar variational autoencoder. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1945–1954.
19. Liu, Q.; Allamanis, M.; Brockschmidt, M.; Gaunt, A. Constrained graph variational autoencoders for molecule design. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 7795–7804.
20. Jin, W.; Barzilay, R.; Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 25–31 July 2018; pp. 2323–2332.
21. De Cao, N.; Kipf, T. MolGAN: An implicit generative model for small molecular graphs. *arXiv* **2018**, arXiv:1805.11973.
22. Pölsterl, S.; Wachinger, C. Adversarial learned molecular graph inference and generation. In Proceedings of the Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, 14–18 September 2020; Proceedings, Part II; Springer: Berlin/Heidelberg, Germany, 2021; pp. 173–189.
23. Prykhodko, O.; Johansson, S.V.; Kotsias, P.C.; Arús-Pous, J.; Bjerrum, E.J.; Engkvist, O.; Chen, H. A de novo molecular generation method using latent vector based generative adversarial network. *J. Cheminform.* **2019**, *11*, 1–13. [[CrossRef](#)] [[PubMed](#)]
24. Zang, C.; Wang, F. MoFlow: An invertible flow model for generating molecular graphs. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual Event, 23–27 August 2020; pp. 617–626.
25. Shi, C.; Xu, M.; Zhu, Z.; Zhang, W.; Zhang, M.; Tang, J. Graphaf: A flow-based autoregressive model for molecular graph generation. *arXiv* **2020**, arXiv:2001.09382.
26. Madhawa, K.; Ishiguro, K.; Nakago, K.; Abe, M. Graphnvp: An invertible flow model for generating molecular graphs. *arXiv* **2019**, arXiv:1905.11600.
27. Wang, Y.; Wu, C.; Herranz, L.; Van de Weijer, J.; Gonzalez-Garcia, A.; Raducanu, B. Transferring gans: Generating images from limited data. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 218–234.
28. Ojha, U.; Li, Y.; Lu, J.; Efros, A.A.; Lee, Y.J.; Shechtman, E.; Zhang, R. Few-shot image generation via cross-domain correspondence. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 10743–10752.
29. Yang, C.; Shen, Y.; Zhang, Z.; Xu, Y.; Zhu, J.; Wu, Z.; Zhou, B. One-shot generative domain adaptation. *arXiv* **2021**, arXiv:2111.09876.
30. Lim, J.; Ryu, S.; Kim, J.W.; Kim, W.Y. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *J. Cheminform.* **2018**, *10*, 1–9. [[CrossRef](#)] [[PubMed](#)]
31. Ma, C.; Zhang, X. GF-VAE: A Flow-based Variational Autoencoder for Molecule Generation. In Proceedings of the CIKM'21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, 1–5 November 2021; Demartini, G., Zuccon, G., Culpepper, J.S., Huang, Z., Tong, H., Eds.; ACM: New York, NY, USA, 2021; pp. 1181–1190. [[CrossRef](#)]
32. Jabbar, A.; Li, X.; Omar, B. A survey on generative adversarial networks: Variants, applications, and training. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–49. [[CrossRef](#)]
33. Jiménez-Luna, J.; Grisoni, F.; Schneider, G. Drug discovery with explainable artificial intelligence. *Nat. Mach. Intell.* **2020**, *2*, 573–584. [[CrossRef](#)]
34. Xu, M.; Cheng, J.; Liu, Y.; Huang, W. DeepGAN: Generating Molecule for Drug Discovery Based on Generative Adversarial Network. In Proceedings of the 2021 IEEE Symposium on Computers and Communications (ISCC), Rhodes Island, Greece, 5–8 September 2021; IEEE: New York, NY, USA, 2021; pp. 1–6.
35. Maziarka, Ł.; Pocha, A.; Kaczmarczyk, J.; Rataj, K.; Danel, T.; Warchoń, M. Mol-CycleGAN: A generative model for molecular optimization. *J. Cheminform.* **2020**, *12*, 1–18.
36. Li, Y.; Zhang, R.; Lu, J.; Shechtman, E. Few-shot Image Generation with Elastic Weight Consolidation. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020.
37. Zhao, M.; Cong, Y.; Carin, L. On Leveraging Pretrained GANs for Generation with Limited Data. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, Virtual Event, 13–18 July 2020; Volume 119, pp. 11340–11351.
38. Yang, C.; Lim, S.N. One-shot domain adaptation for face generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 5921–5930.
39. Altae-Tran, H.; Ramsundar, B.; Pappu, A.S.; Pande, V. Low data drug discovery with one-shot learning. *ACS Cent. Sci.* **2017**, *3*, 283–293. [[CrossRef](#)] [[PubMed](#)]

40. Lv, Q.; Chen, G.; Yang, Z.; Zhong, W.; Chen, C.Y.C. Meta Learning With Graph Attention Networks for Low-Data Drug Discovery. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, 1–13. [[CrossRef](#)]
41. Xu, Z.; Wauchope, O.R.; Frank, A.T. Navigating chemical space by interfacing generative artificial intelligence and molecular docking. *J. Chem. Inf. Model.* **2021**, *61*, 5589–5600. [[CrossRef](#)]
42. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A.C. Improved training of wasserstein gans. *Adv. Neural Inf. Process. Syst.* **2017**, 30.
43. Xu, Y.; Shen, Y.; Zhu, J.; Yang, C.; Zhou, B. Generative hierarchical features from synthesizing images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4432–4442.
44. Dara, S.; Tumma, P. Feature extraction by using deep learning: A survey. In Proceedings of the 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 29–31 March 2018; IEEE: New York, NY, USA, 2018; pp. 1795–1801.
45. Sterling, T.; Irwin, J.J. ZINC 15—ligand discovery for everyone. *J. Chem. Inf. Model.* **2015**, *55*, 2324–2337. [[CrossRef](#)]
46. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
47. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, 32.
48. Landrum, G. RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum* **2013**, *8*, 31.
49. Hsu, J. COVID-19: What now for remdesivir? *BMJ* **2020**, 371. [[CrossRef](#)] [[PubMed](#)]
50. McDonald, E.G.; Lee, T.C. Nirmatrelvir-ritonavir for COVID-19. *CMAJ* **2022**, *194*, E218. [[CrossRef](#)] [[PubMed](#)]
51. Jorgensen, S.C.; Tse, C.L.; Burry, L.; Dresser, L.D. Baricitinib: A review of pharmacology, safety, and emerging clinical experience in COVID-19. *Pharmacother. J. Hum. Pharmacol. Drug Ther.* **2020**, *40*, 843–856. [[CrossRef](#)]
52. Markowski, M.C.; Tutrone, R.; Pieczonka, C.; Barnette, K.G.; Getzenberg, R.H.; Rodriguez, D.; Steiner, M.S.; Saltzstein, D.R.; Eisenberger, M.A.; Antonarakis, E.S. A Phase Ib/II Study of Sabizabulin, a Novel Oral Cytoskeleton Disruptor, in Men with Metastatic Castration-resistant Prostate Cancer with Progression on an Androgen Receptor-targeting Agent. *Clin. Cancer Res.* **2022**, *13*, 2789–2795. [[CrossRef](#)]
53. Singh, A.K.; Singh, A.; Singh, R.; Misra, A. Molnupiravir in COVID-19: A systematic review of literature. *Diabetes Metab. Syndr. Clin. Res. Rev.* **2021**, *15*, 102329. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.