Software Quality Journal (2024) 32:791–819
https://doi.org/10.1007/s11219-024-09671-7

RESEARCH

Quantitative evaluation of molecular generation performance of graph-based GANs

Jinli Zhang[1] · Zhenbo Wang[1] · Zongli Jiang[1] · Man Wu[2] · Chen Li[3] · Yoshihiro Yamanishi[3]

Accepted: 1 April 2024 / Published online: 25 April 2024

Abstract

Deep generative models have been widely used in molecular generation tasks because they can save time and cost in drug development compared with traditional methods. Previous studies based on generative adversarial network (GAN) models typically employ reinforcement learning (RL) to constrain chemical properties, resulting in efficient and novel molecules. However, such models have poor performance in generating molecules due to instability in training. Therefore, quantitative evaluation of existing molecular generation models, especially GAN models, is necessary. This study aims to evaluate the performance of discrete GAN models using RL in molecular generation tasks and explore the impact of different factors on model performance. Through evaluation experiments on QM9 and ZINC datasets, the results show that noise sampling distributions, training epochs, and training data volumes can affect the performance of molecular generation. Finally, we provide strategies for stable training and improved performance for GAN models.

Keywords Quantitative evaluation · Molecular generation · Generative adversarial network · Reinforcement learning

1 Introduction

Studies on new drug development have always been subject to the limitations of traditional research methods in the field of medicine. Traditional methods require significant financial and time investments (Sarpong et al., 2023), relying on continuous experimentation and optimization to identify suitable compounds. The inefficiency of this approach can be frustrating (Walker, 1998). Additionally, the discrete and complex nature of chemical space makes

✉ Man Wu
wu.man.wi5@am.ics.keio.ac.jp

✉ Chen Li
li.chen.z2@a.mail.nagoya-u.ac.jp

[1] Faculty of Information Technology, Beijing University of Technology, Beijing, China

[2] Department of Information and Computer Science, Keio University, Yokohama, Japan

[3] Graduate School of Informatics, Nagoya University, Nagoya, Japan

it even more challenging for traditional methods to find effective drug candidates. These challenges have prompted researchers to seek more efficient and innovative approaches to accelerate the drug development process (Kim et al., 2016; Li & Yamanishi, 2023). In recent years, deep learning has been widely used in stock price prediction (Zhang et al., 2019), software reliability (Chen et al., 2022), recommendation systems (An et al., 2023; Li et al., 2018, 2023), and medical systems (Jiang et al., 2022; Zhao et al., 2022). The development of deep learning provides effective solutions to drug generation research challenges (Bagal et al., 2021; De Cao & Kipf, 2018; Li et al., 2021). Compared to traditional methods, deep learning approaches offer several advantages in accelerating the drug discovery process (Rifaioglu et al., 2021). They enable rapid screening and prediction of molecular properties and activities, allowing researchers to identify potential drugs from a large pool of candidate compounds. Deep learning methods also provide valuable information to guide experimental design and optimization efforts. Furthermore, deep learning-based methods automate and optimize labor-intensive experimental processes, enhancing research and development efficiency while minimizing resource wastage. By leveraging predictions and simulations, the number of trial-and-error experiments in the laboratory can be reduced, leading to shorter development cycles and cost savings.

In deep learning approaches, the generative models based on Variational Autoencoders (VAEs) (Gómez-Bombarelli et al., 2018; Kingma & Welling, 2013; Shi et al., 2020) capture the underlying patterns and structures of molecules, and then reconstruct and decode them to generate desired new molecules. However, VAEs have limitations in that their main objective is to minimize the difference between the generated outputs and the input data, which may prioritize the generation of data closely similar to the real data rather than truly learning how to generate novel and diverse data. This emphasis on minimizing reconstruction error can hinder the model's ability to explore and generate novel and unique data points. Therefore, VAEs may have shortcomings in fully capturing the underlying distribution and generating outputs beyond the scope of the training dataset. The inspiration behind the diffusion model comes from physics. In physics, gas molecules diffuse from areas of high concentration to areas of low concentration, similar to the loss of information due to interference from noise. Therefore, by introducing noise and then attempting to denoise it, we can reconstruct the underlying data itself. Over a period of time and multiple iterations, the model learns to generate new samples given some noisy input. The principle is to learn the information decay caused by noise and then use the learned patterns to generate samples. This concept also applies to latent variables as it aims to learn the distribution of noise rather than the distribution of data. Diffusion models have recently found applications in molecular generation tasks as well.

An alternative approach is based on generative adversarial networks (GANs) (Goodfellow et al., 2020; Guimaraes et al., 2017; Kadurin et al., 2017; Sanchez-Lengeling et al., 2017; You et al., 2018; Yu et al., 2017), comprising a generator and a discriminator. The generator produces samples and forwards them to the discriminator for assessment, with the discriminator providing judgments based on the quality of the generated samples. The generator adjusts its generation strategy based on feedback from the discriminator to enhance sample quality. The discriminator continuously refines its discrimination accuracy using the performance of generated and real samples. Through iterative training, the generator and discriminator progressively improve their abilities, ultimately reaching a dynamic equilibrium. The processes of deep generative models are depicted in Fig. 1. In the context of molecule generation, reinforcement learning (RL) (Williams, 1992) is commonly utilized to impose constraints on generated molecules, optimizing their properties. This approach offers the advantage of enabling the generated molecules to possess desired properties. However, GANs are susceptible to mode collapse, and

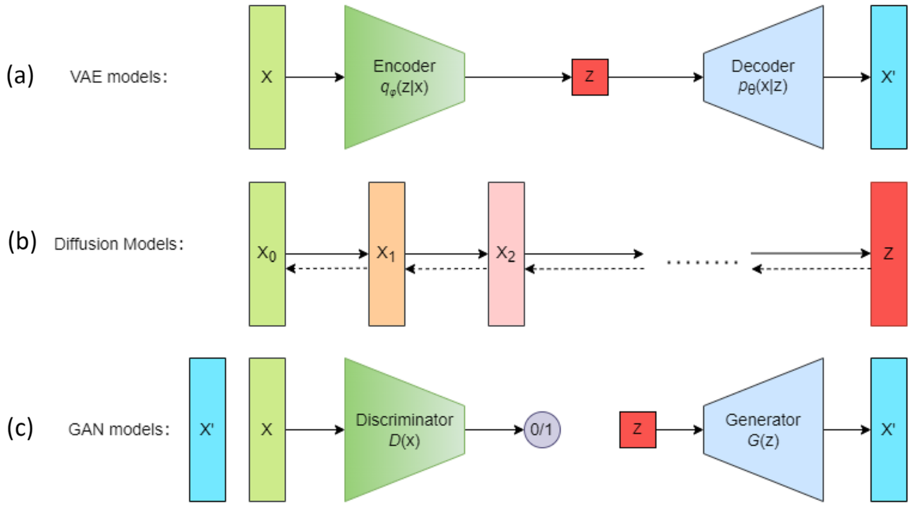


Fig. 1 Three commonly used deep generative models: VAE models, diffusion models, and GAN models. **a** VAE models consist of two parts: the encoder and the decoder. The encoder maps input data to a latent variable representation in the latent space, while the decoder reconstructs the original input data based on these latent variables. **b** Diffusion models mainly consist of a forward process and a backward process. The forward process continuously adds noise to the input data, and as time steps approach infinity, it eventually becomes pure noise. The backward process is the denoising and recovery process. It is also the process of generating the target. **c** GAN models consist of two parts: the generator and the discriminator. The goal of GAN is to train the generator to generate realistic data samples, while the discriminator tries its best to distinguish the differences between the samples generated by the generator and real samples

when combined with RL, the model becomes more unstable. Prolonged training may lead to the generation of molecules limited to only a few types. Different models exhibit distinct advantages and limitations. It is crucial in the task of molecule generation to accurately evaluate a model, identify its strengths, and assess its weaknesses. Only through precise evaluation can we leverage a model's strengths while mitigating its negative effects, thereby generating molecules that better align with our expectations.

In contrast to the conventional GAN model, the discrete GAN model is primarily utilized for processing discrete data types such as text or categorical labels. Molecules, being a form of highly sparse data, are well-suited for analysis using the discrete GAN model. The evaluation of GAN quality is a multifaceted task that typically involves considering various factors. It is imperative to assess the quality of generated samples by comparing them to real samples. Moreover, evaluating distribution matching is crucial as GANs strive to align the distribution of generated samples with that of real samples. When addressing specific tasks like molecule generation, it becomes essential to assess the diversity, effectiveness, and other relevant metrics associated with the generated molecules. Specifically in the context of molecule generation, an evaluation of the distribution of generated molecules in comparison to the original dataset is necessary. This is commonly achieved by calculating diverse chemical metrics to compare the distributions between the generated molecules and the original data. A closer similarity in the distributions of these metrics indicates a more effective learning of distribution characteristics by the generator. Furthermore, beyond the comprehensive evaluation metrics for molecule generation mentioned above, assessing individual generated samples plays a pivotal role in molecular generation tasks. The primary objective of such tasks is to generate molecules that exhibit similar properties to the training data while also demonstrating novel characteristics. Typically, effectiveness, uniqueness, and novelty are employed for evaluation purposes. Effectiveness gauges the model's capacity to understand the fundamental structure of molecules, while uniqueness helps determine if the model solely captures partial features, providing insights into mode collapse. Novelty, on the other hand, evaluates the model's capability to explore the chemical space.

This study primarily focuses on evaluating the performance of discrete GAN models in molecular generation tasks. These types of models maintain the structure of classical GAN models, consisting of a generator and a discriminator, with the addition of RL-based optimization. The main contributions of this study are as follows:

- This study primarily focuses on evaluating deep learning models for drug design, particularly graph-based RL-driven GANs.
- We evaluate the discrete GAN models by examining their generated molecular properties, as well as their overall effectiveness, uniqueness, and novelty, among other evaluation metrics.
- To evaluate the models, we use two datasets and conduct extensive experiments by selecting multiple factors that influence the model's generative capability. These experiments further contribute to assessing the model's generative ability.

2 Related work

2.1 Performance evaluation using VAEs

Previously, in molecular generation tasks, methods based on VAE models have received considerable attention. CharVAE (Gómez-Bombarelli et al., 2018) is a character-level VAE that has demonstrated good performance in molecular generation tasks using SMILES data. GramVAE (Kusner et al., 2017) is a grammar-based VAE capable of generating diverse molecules that adhere to grammar rules while also possessing interpretability. Compared to other molecular generation methods, GramVAE exhibits better control over the structure and properties of generated molecules. GraphVAE (Simonovsky & Komodakis, 2018) is a type of VAE specifically designed for generating and learning representations of graph-structured data. In the field of molecular chemistry, GraphVAE has been applied to molecular generation and design, allowing for the generation of diverse compounds with favorable properties. In social network analysis, GraphVAE is capable of discovering community structures and identifying key nodes. In recommendation systems, GraphVAE can provide personalized recommendations. Overall, GraphVAE demonstrates excellent performance in generating and learning graph-structured data. It can generate diverse and reasonable graph samples and can be controlled and optimized according to specific tasks. GraphAF (Shi et al., 2020) is a flow-based autoregressive model utilized for molecular graph generation. It is composed of an autoregressive flow model and a graph neural network. GraphAF has the capability to generate diverse and reasonable molecular structures, overcoming the limitations of traditional rule-based or search-based models that struggle to generate diverse molecules. GraphAF incorporates a control mechanism, enabling users to generate molecular graphs with specific properties by setting specific conditions or goals. Furthermore, GraphAF leverages flow transformation techniques, accelerating the speed of molecular generation, and demonstrating high efficiency and scalability.

2.2 Performance evaluation using diffusion models

Diffusion models have recently found applications in molecular generation and drug discovery tasks as well. The Molecular Diffusion Model (MDM) (Huang et al., 2023) addresses the challenges of capturing interatomic interaction potentials and lack of diversity. MDM combines enhanced encoding of varying strength of interatomic forces with a dual-equivalent encoder. To enhance the diversity of generated samples, they introduce latent variables in the diffusion and generation processes, leading to superior performance on drug-like datasets compared to state-of-the-art (SOTA) models. GEOLDM (Xu et al., 2023) achieves impressive performance in multiple molecular generation benchmark tests by capturing critical roto-translational equivalence constraints through the construction of a point-structured latent space with invariant scalars and equivariant tensors. In order to make the generation of molecules more flexible and controllable, EEGSDE (Bao et al., 2022) adopts an equivariant SDE as the framework, guided by a meticulously designed energy function. The generated molecules, at each step, are tailored towards quantum properties, molecular structures, and even their combinations. Additionally, gradient descent is applied to the energy function to encourage the generated molecules to have low energy. Guided by the energy function, EEGSDE is more favorable than the current SOTA molecular structure model, EDM (Hoogeboom et al., 2022), for applications such as drug discovery or material exploration. GCDM (Morehead & Cheng, 2023) proposed a novel approach for generating three-dimensional molecular structures using a diffusion process that takes into account the molecular geometric shape. The model encodes the two-dimensional structure of the molecule using graph convolutional neural networks and then generates a three-dimensional structure compatible with the input two-dimensional structure through diffusion. This method has been demonstrated to create high-quality 3D structures that compete with SOTA methods while requiring fewer computational resources. It shows potential application prospects in drug discovery and materials design.

2.3 Performance evaluation using GANs

In contrast to VAE models, GAN models are comprised of two primary elements: a generator network and a discriminator network. The generator network accepts random noise as input and produces synthetic data samples with the objective of generating molecules that closely resemble real SMILES strings. On the other hand, the discriminator network functions as a binary classifier, attempting to differentiate between real and generated molecules. Unlike traditional GAN models, ORGAN (Guimaraes et al., 2017) leverages RL to optimize using reward functions. This allows the ORGAN model to achieve better results in sequence generation tasks and specifically optimize for certain objectives, such as generating high-quality and diverse sequence data. This allows the ORGAN model to achieve better results in sequence generation tasks and optimize specifically for certain objectives, such as generating high-quality and diverse sequence data. MolGAN (De Cao & Kipf, 2018) is a molecular generation model that utilizes deep deterministic policy gradients (DDPG) (Lillicrap et al., 2015) and an improved Wasserstein GAN (WGAN) to generate molecules. MolGAN exhibits high effectiveness, meaning that the generated molecules are chemically valid and can be successfully synthesized and utilized. Additionally, MolGAN is capable of generating diverse chemical structures by producing multiple types of molecules. However, due to the issue of mode collapse in WGAN, the generated molecules may lack sufficient uniqueness. MolCycleGAN (Maziarka et al., 2020) leverages JT-VAE (Jin et al., 2018) to learn the topological information of molecules and then utilizes

CycleGAN (Zhu et al., 2017) to learn the physicochemical properties of molecules. This enables the generation of molecules with desired properties. DNMG (Song et al., 2023) is a GAN model that incorporates transfer learning, and the molecules it generates have better binding affinity and improved physicochemical properties towards target proteins.

3 Model description

The architecture of the model, as depicted in Fig. 2, follows the fundamental structure of RL-based discrete GAN models. These models typically consist of three components: a generator G_θ, a discriminator D_ϕ, and a reward network $\hat{R}_\psi$. To generate valid molecules, the generator samples from a prior distribution or noise and generates an annotated graph G to represent a molecule. The nodes and edges of G are annotated with information regarding atom types and bond types, respectively. The discriminator is trained to differentiate between samples from the dataset and those generated by the generator. By employing an improved version of WGAN, the generator G_θ and discriminator D_ϕ are trained to enable the generator to match the empirical distribution and produce valid molecules.

The role of the reward network in discrete GAN models is particularly significant and cannot be overlooked. It serves to approximate the reward function of a sample and

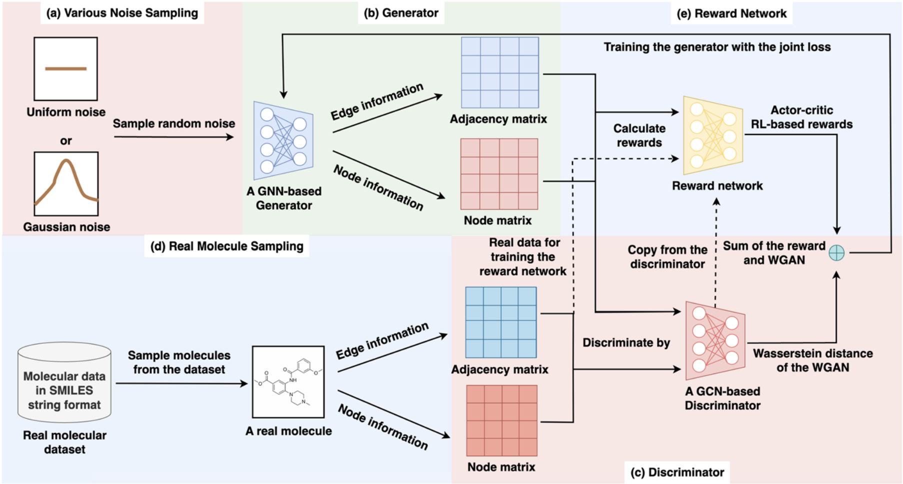

(a) Various Noise Sampling
(b) Generator
(e) Reward Network
Training the generator with the joint loss
Uniform noise
or
Gaussian noise
Sample random noise
A GNN-based Generator
Edge information
Node information
Adjacency matrix
Node matrix
Calculate rewards
Actor-critic RL-based rewards
Reward network
(d) Real Molecule Sampling
Real data for training the reward network
Copy from the discriminator
Sum of the reward and WGAN
Molecular data in SMILES string format
Real molecular dataset
Sample molecules from the dataset
A real molecule
Edge information
Node information
Adjacency matrix
Node matrix
Discriminate by
A GCN-based Discriminator
Wasserstein distance of the WGAN
(c) Discriminator

Fig. 2 Overview architecture of Graph-based RL-driven GANs. **a** The generator samples noise from a Gaussian or uniform distribution. **b** After obtaining the samples, the generator is processed by a GNN, which generates two matrices: the adjacency matrix representing edge information and the node matrix representing node information. **c** The discriminator receives the matric generated by the generator and the matric from the real dataset and calculates the Wasserstein distance between the two data distributions. **d** The training data is initially sampled from the dataset and obtained in the form of SMILES strings. Then, an external tool is used to convert the string-formatted data into graph-structured data, which is represented by two matrices respectively indicating edge information and node information. **e** The reward network receives data generated by the generator and uses the actor-critic algorithm to compute rewards. These rewards, combined with the Wasserstein distance from the discriminator, form the joint loss used to train the generator. Additionally, the reward network also receives data from the dataset and trains itself by comparing it against real scores

employs RL techniques to optimize molecule generation, addressing the challenges posed by non-differentiable metrics. Unlike the discriminator, the reward network assigns scores to both the dataset and generated samples, based on specific properties. Its primary function is to align the assigned scores for each molecule with the scores provided by external software, effectively assigning rewards to every generated molecule. Notably, when an invalid molecule is inputted into the reward network, it is assigned a score of zero, as the graph representation cannot depict a compound.

3.1 Generator

In this model, we employ a generator based on the graph neural network architecture for evaluation. The generator consists of a 3-layer MLP with hidden units of [128; 256; 512] respectively, and uses hyperbolic tangent (tanh) as the activation function. The generator typically samples from noise to generate molecular graphs based on the sampled noise input, which includes node feature matrices and adjacency matrices. The final component of the generator is a multi-layer perceptron, enabling it to simultaneously predict an entire graph, thereby improving the speed of molecule generation and property optimization. We impose a limitation on the number of nodes in the generated graph to a finite range. For each input noise z, G_θ produces two densely connected and continuous objects: $\boldsymbol{X} \in \mathbb{R}^{N\times T}$ to specify atom types, and $\boldsymbol{A}^{N\times N\times Y}$ to specify bond types. The variables $\mathbf{X}$ and $\mathbf{A}$ can be probabilistically interpreted, where each node and edge type is represented by probabilities derived from categorical distributions over types. To create a molecule, we sample from these two adjacency matrices to obtain discrete objects: $\tilde{\boldsymbol{X}}$ and $\tilde{\boldsymbol{A}}$. Then, these two discrete graph structures are passed to the discriminator and reward network, and the generator loss is computed as:

$$L_{G_\theta} = E_{z\sim p_z(z)} \log(1 - D_\phi(G_\theta(z))). \tag{1}$$

Here, z represents the data sampled from random noise, and $G_\theta(z)$ represents the molecular representation generated by the generator. For the generator, its objective is to minimize the output values when the generated data $G_\theta(z)$ is evaluated by the discriminator.

3.2 Discriminator and reward network

The discriminator and reward network parts of the model under evaluation both adopt the same relation-based graph convolutional neural network (Schlichtkrull et al., 2018) architecture (with non-shared parameters). This architecture supports graphs with multiple edge types. Although the two networks receive the same data, their functions are different. Both networks utilize a relational GCN encoder comprising two layers and [64; 32] hidden units, respectively, for processing the input graphs. Following this, we calculate a 128-dimensional graph-level representation, which is subsequently processed by a 2-layer MLP with dimensions [128; 1] and tanh activation function for the hidden layer. In the reward network, we additionally apply a sigmoid activation function to the output.

Generally, a GCN-based model takes graph-structured data as input. In the task of molecule generation, each atom in the molecule corresponds to a node i in the graph structure $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$, and its relevant feature is represented as Xi. The features of these atoms are combined into a feature matrix $\mathbf{X}$ of size $N \times D$ (where N represents the number of nodes, and D

represents the number of input features), which is then jointly represented with the adjacency matrix **A** to form a graph structure. Subsequently, a node-level output Z is obtained, which is an $N \times F$ feature matrix (where F represents the number of output features for each node). Finally, graph-level outputs can be achieved through certain types of pooling operations. The graph convolution process can be expressed by the following formula:

$$f(\boldsymbol{H}^{(l+1)}, \boldsymbol{A}) = \sigma\left(\hat{\boldsymbol{D}}^{-\frac{1}{2}} \hat{\boldsymbol{A}} \hat{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{H}^{(l)} \boldsymbol{W}^{(l)}\right). \tag{2}$$

Here, $\boldsymbol{H}^{(l+1)}$ is the feature representation matrix for the (l+1)-th layer, $\boldsymbol{A}$ is the adjacency matrix, $\boldsymbol{D}$ is the degree matrix the diagonal elements are the degree of each node, and $\boldsymbol{W}^{(l)}$ the weight matrix for the l–th layer. The feature matrix $\boldsymbol{H}^{(l)}$ is first subjected to a linear transformation $\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}$, and then the adjacency matrix $\boldsymbol{A}$ is used to represent the connection between nodes. The operation $\hat{\boldsymbol{D}}^{-\frac{1}{2}} \hat{\boldsymbol{A}} \hat{\boldsymbol{D}}^{-\frac{1}{2}}$ is used to obtain the information transmission between nodes, and finally the (l+1)-th layer feature representation matrix $\boldsymbol{H}^{(l+1)}$ is obtained through an activation function. When using relational GCN, the propagation method at each layer for nodes is as follows:

$$\boldsymbol{h}_i^{(\ell+1)} = \tanh\left(f_s^{(\ell)}\left(\boldsymbol{h}_i^{(\ell)}, \boldsymbol{x}_i\right) + \sum_{j=1}^{N} \sum_{y=1}^{Y} \frac{\tilde{\boldsymbol{A}}_{ijy}}{|\mathcal{N}_i|} f_y^{(\ell)}\left(\boldsymbol{h}_j^{(\ell)}, \boldsymbol{x}_j\right)\right), \tag{3}$$

where $\boldsymbol{h}_i^{(\ell)}$ is the signal of the node i at layer, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the feature representations of node i and node j, respectively. $f_s^{(\ell)}$ is a linear transformation function that acts as a self-connection between layers. The model further utilized an edge type-specific affine function $f_y^{(\ell)}$ for each layer. $\mathcal{N}_i$ denotes the set of neighbors for node i. The normalization factor $1/|\mathcal{N}_i|$ ensures that activations are on a similar scale irrespective of the number of neighbors.

After passing through multiple layers of graph convolutions (Li et al., 2015), we combined the embeddings of nodes to form a vector representation that represents the entire graph:

$$\boldsymbol{h}_{\mathcal{G}} = \tanh\left(\sum_{v \in \mathcal{V}} \sigma\left(i\left(\boldsymbol{h}_v^{(L)}, \boldsymbol{x}_v\right)\right) \odot \tanh\left(j\left(\boldsymbol{h}_v^{(L)}, \boldsymbol{x}_v\right)\right)\right), \tag{4}$$

where σ is the logistic sigmoid function, i, and j are Multi-Layer Perceptrons (MLPs) with a linear output layer and $\odot$ denotes element-wise multiplication. Then, $\boldsymbol{h}_{\mathcal{G}}$ is a vector representation of the graph $\mathcal{G}$ and it is further processed by an MLP to produce a graph-level scalar output$\in (-\infty, +\infty)$ for the discriminator and in range of [0, 1] for the reward network. The loss function of the discriminator, in comparison to the generator, is as follows:

$$L_{D_\phi} = E_{x \sim p_{data}(x)} \log(D_\phi(x)). \tag{5}$$

Here, the symbol x represents the samples from the real dataset, thus this formula aims for the discriminator to assign higher scores to the real s. By combining the formulas of the discriminator and the generator, the loss function of GAN as follows:

$$L = E_{x \sim p_{data}(x)} \log(D(x)) + E_{z \sim p_z(z)} \log(1 - D(G(z))). \tag{6}$$

When the generated data can better deceive the discriminator, the value of the loss function is smaller. Conversely, when the discriminator has a stronger ability to distinguish, the value of the loss function is larger. After a series of iterations and games, the system eventually reaches an equilibrium point.

3.3 Reinforcement learning algorithm

In this study, we evaluated a discrete GAN model. The generator is theoretically capable of learning the distribution of the training data and generating samples similar to real data. In molecule generation tasks, it is essential to ensure that generated molecules comply with chemical rules while possessing specific chemical properties.

To address this issue, many models use RL (Williams, 1992) methods. RL is a machine learning method used to train an agent to learn an optimal strategy by interacting with the environment. In molecule generation tasks, the generator is considered an agent that selects an action (generating a molecule) based on the current state (prior samples) and updates its policy based on rewards (chemical properties) given by the environment. The goal of RL is to learn to generate high-quality molecules by maximizing the cumulative reward. A reward function is designed using RL to evaluate the chemical properties of the generated molecules and adjust the generator's policy based on the evaluation results. This process can be implemented using discrete RL algorithms such as Q-learning or Policy Gradient. By iteratively training the generator and continuously optimizing the reward function and policy, we can ensure that the generated molecules comply with chemical rules and possess specific chemical properties.

In RL, a policy defines the decision-making process of an agent. A stochastic policy entails the agent probabilistically selecting from a range of possible actions while in a specific state, allowing for variability in actions even within the same state. The utilization of a stochastic policy offers the benefit of exploring new strategies by iteratively making random selections to uncover improved policies. Conversely, a deterministic policy involves the agent choosing a fixed action when in a particular state, as opposed to selecting actions randomly based on a probability distribution. Deterministic policies are often preferred when dealing with continuous action spaces due to their capacity to retain past experiences and progress towards more optimal directions.

To introduce constraints, we implemented the DDPG algorithm for evaluating the model. This off-policy Actor-Critic algorithm is particularly well-suited for handling high-dimensional action spaces when generating graphs. In the strategy outlined in this paper, the generator G_θ, acting as an agent, takes prior sample z as input, rather than the environmental state s typically used in reinforcement learning. Consequently, the molecular graph generated by the generator is treated as an action ($a = \mathcal{G}$) and fed into the reward network. Furthermore, since the action is solely determined by the generated graph $\mathcal{G}$, there is no need to evaluate the quality of state-action pairs. Based on this, we utilize a trainable and differentiable reward function $\hat{R}_\psi(\mathcal{G})$ to predict real-time rewards. Additionally, we introduce an external evaluation system that provides actual rewards for the actions generated by the generator, namely the molecular graph. The reward network is trained using mean squared error. Exploiting this, we train the generator by maximizing the predicted rewards, which are differentiable and provide gradients towards the desired metrics. In adversarial training, the loss function L_{RL} is defined as follows:

$$L_{RL} = \mathbb{E}\left[\left(R_{Real} - \hat{R}_\psi(\mathcal{G})\right)^2\right], \tag{7}$$

where R_{Real} represents the real reward for the current molecule provided by an external system, and $\hat{R}_\psi(\mathcal{G})$ is the reward predicted by the reward network.

4 Performance evaluation

In order to comprehensively evaluate the generation performance of the model, this study mainly calculates two types of sample evaluation indicators, which are sample statistics and sample attributes, and evaluates the quality of the model's generation by comparing it with the training data.

4.1 Evaluation metrics for sample statistics

For the statistical evaluation of generated samples (Samanta et al., 2018), we mainly calculate the following four indicators: Validity, uniqueness, novelty, and diversity.

- **Validity** refers to the proportion of valid molecules in a batch of data generated by the generator, as determined by external software, relative to the total number of sampled data. The calculation formula is as follows:

$$S_{Validity} = \frac{N_{Valid}}{N_{Sample}} \tag{8}$$

  Here, the symbol N_{Sample} represents the number of sampled data from the generator, and the symbol N_{Valid} represents the number of valid molecules generated by the generator. By evaluating the value of validity, we can assess how well the model has learned the basic structure of molecules in the training data. Only by conducting statistical analysis on validity can the subsequent evaluation process be meaningful.
- **Uniqueness** is calculated by determining the ratio between the number of unique samples and the total number of valid samples. The calculation formula is as follows:

$$S_{Uniqueness} = \frac{N_{Valid} - N_{Repeated}}{N_{Valid}} \tag{9}$$

  Here, the symbol $N_{Repeated}$ represents the number of repeated samples in the generated valid samples, and $S_{Uniqueness}$ represents the proportion of unique samples that exist in the generated valid samples. Uniqueness can be used to verify whether the model has accurately learned the distribution of the training data. For GAN models, it can be used to evaluate whether mode collapse has occurred.
- **Novelty** measures the proportion of unique valid samples that are not present in the training dataset. It can evaluate a model's exploratory ability, which is an important criterion for generating new molecules. The calculation formula is as follows:

$$S_{Novelty} = \frac{N_{Unique-Orignal}}{N_{Unique}} \tag{10}$$

  Here, symbol $N_{Unique-Orignal}$ represents the number of unique samples in the generated data that are not present in the training data, while symbol N_{Unique} represents the number of unique and valid molecules.
- **Diversity** is usually defined as the average Tanimoto distance between the Morgan fingerprints of newly generated molecules (Rogers & Tanimoto, 1960).

4.2 Evaluation metrics for property optimization

Evaluating the model based on the attributes of generated molecules is an important task, especially when the goal is to generate molecules with specific attributes. We evaluate the model using some commonly used molecular attribute evaluation methods, specifically Drug-likeness (QED), Logarithm of the partition coefficient (LogP), and Synthesizability (SA), for assessing the generated molecules.

- **Quantitative Estimation of Drug-likeness (QED)** (Bickerton et al., 2012) is a metric used in drug discovery and medicinal chemistry to assess the drug-likeness of small molecules. QED is a numerical value that quantifies the similarity of a given compound to known drugs based on several molecular descriptors. QED is intuitive, transparent, easily implementable in many practical settings, and allows ranking of compounds based on their relative scores. Extensive work in drug design has shown the potential of QED for assessing the drug-like properties of molecules targeted at specific molecular targets, making it a key assessment method for drug similarity.
- **Logarithm of the partition coefficient (LogP)** (Comer & Tam, 2001), is a parameter used to evaluate and describe the lipophilicity of a compound. LogP refers to the logarithm of the partition coefficient, which represents the distribution behavior of a molecule between a nonpolar solvent (e.g., oil) and water at the interface. A higher LogP value indicates a molecule that is more hydrophobic, while a lower LogP value indicates a molecule that is more hydrophilic.
- **Synthesizability (SA)** (Ertl & Schuffenhauer, 2009), is a commonly used evaluation metric to assess the synthetic feasibility of generated molecules. The SA metric measures the complexity and difficulty of a molecule's chemical synthesis. It helps determine whether there are suitable synthetic routes to produce the generated molecule and evaluates its achievability. The SA metric is of significant importance in drug design and compound optimization. It aids in screening compounds with good synthetic potential and feasibility, thereby improving the efficiency of drug development.

5 Numerical experiments

We conducted a series of experiments on the established benchmark using two chemical datasets, ZINC and QM9. Additionally, we explored and investigated the effects of various factors (such as noise sampling distributions, training epochs, and training data volumes) on the performance of the model.

5.1 Datasets

In the evaluation of the model, we utilized the ZINC (Irwin et al., 2012) and QM9 (Ramakrishnan et al., 2014) datasets.

- **ZINC dataset** is a widely used and publicly available chemical database primarily used for drug discovery and chemical research. This dataset contains a large number of molecular data, and each molecule has associated chemical properties. The ZINC dataset covers various types of atoms. Specifically, it includes common atomic elements found in organic compounds, such as carbon (C), hydrogen (H), oxygen (O), nitrogen (N), sulfur (S), and more. In terms of the number of molecules in the dataset, ZINC contains approximately 250,000 molecular data points. Each molecule has its specific structure, chemical properties, and activity information. These pieces of information can be used to evaluate the potential effects of drug candidate compounds, compare the similarity between different compounds, and study the relationship between molecular structure and properties.

- **QM9 dataset** is a subset of the extensive GDB-17 (Ramakrishnan et al., 2014) chemical database, which consists of a remarkable 166.4 billion molecules. Within the QM9 dataset, there are 133,885 organic compounds that contain a maximum of nine heavy atoms, including carbon (C), oxygen (O), nitrogen (N), and fluorine (F). QM9 provides accurate and detailed quantum chemical features specifically tailored for small organic molecules. The QM9 dataset is highly valuable for conducting theoretical calculations and property prediction research on organic molecules. It provides abundant information that can be utilized for developing molecule models, designing new organic materials, and optimizing catalysts. Additionally, the QM9 dataset has been extensively employed in the development of machine learning and deep learning methods, enabling efficient prediction of organic molecular properties and material discovery.

For the two datasets, we randomly extracted one-tenth of the data from each dataset to form new datasets for evaluation. We named these new datasets as ZINC_25k and QM9_13k, respectively. This approach serves two purposes: firstly, it reduces the training time by working with smaller datasets, and secondly, it enables us to compare the results with the original datasets for a better assessment of the model's performance. Overall, the QM9 dataset focuses on quantum chemical research of organic molecules, providing abundant molecular property information suitable for theoretical calculations and property prediction. On the other hand, the ZINC dataset has a broader scope, encompassing various types of molecules, and is primarily used for drug discovery and chemical research.

5.2 Hyperparameters

For the structure of each component in the model under evaluation, we maintain consistency across all experiments. For the ZINC dataset, we set the following parameters: $N = 40$ represents the maximum node count, which corresponds to the length of the molecule. $T = 9$ represents the number of different atom types, and $Y = 4$ represents the various bond types between atoms (single, double, triple, and no bond). This ensures that all molecules in the dataset can be adequately represented. In the case of the QM9 dataset, we set $N = 9$ and $T = 5$, as compared to ZINC, QM9 data is shorter with fewer types of atoms.

The generator samples from a standard normal distribution and then passes through a three-layer MLP with hidden unit sizes set to [128, 256, 512]. Subsequently, it goes through the tanh activation function. Finally, the last layer is linearly projected to match the dimensions of $\boldsymbol{X}$ and $\boldsymbol{A}$ and normalized using softmax in the last dimension $\mathrm{softmax}(\boldsymbol{x})_i = \exp(\boldsymbol{x}_i) / \sum_{i=1}^{D} \exp(\boldsymbol{x}_i)$.

For both the discriminator and the reward network, we employ relational graph convolutional networks to process the input graphs, with hidden unit sizes set to [64, 32]. The final two-layer MLP has dimensions [128, 1]. Specifically, for the reward network, we activate the output using a sigmoid function.

We have set the following standards for evaluating the model: the sampling method is set to random sampling, the number of training epochs is set to 150, and the training data consists of one-tenth of the original dataset, which is mentioned earlier as ZINC_25k and QM9_13k. Then, we evaluate the model by varying its parameters based on these standards.

5.3 Experimental results

Evaluation results under different sampling methods We evaluated the model using two different sampling methods: normal sampling and uniform sampling, on two subsets of ZINC datasets and QM9 datasets. Normal sampling means sampling from a normal distribution, while uniform sampling means sampling from a uniform distribution. The training epochs are set to 150. For each sampling method, we evaluate the performance of the model with sample sizes ranging from 100 to 1000. All data highlighted in bold in the tables represent the optimal results for that indicator.

Table 1 presents the evaluation results using different sampling methods on the ZINC_25k subset. Based on the Validity metrics, the model consistently generates molecules with a validity rate above 72.30% under the normal sampling method, outperforming the uniform sampling method. This suggests that the model is more capable of generating valid molecules under normal sampling. Furthermore, as the sample size increases, the validity of the generated molecules does not decrease, indicating that the model has learned the basic structure of molecules. Regarding uniqueness, although the metric is higher for uniform sampling with a sample size of 1000, in reality, the normal sampling method generates a greater number of unique and valid molecules. As for novelty and diversity, while the model performs well on both methods, overall, it exhibits better performance on the normal sampling method.

Table 2 presents the evaluation results using different sampling methods on the QM9_13k subset. The experimental process is the same as that used for the ZINC_25k

Table 1 Evaluation results of different sampling methods on the ZINC_25k subset

Sampling method	Sampling size	Validity ↑	Uniqueness ↑	Novelty ↑	Diversity ↑
Normal	100	79 (79.00%)	**69 (87.34%)**	**67 (97.10%)**	0.94
	300	238 (79.33%)	193 (81.09%)	185 (95.85%)	0.94
	500	396 (79.20%)	294 (74.24%)	276 (93.88%)	0.94
	700	**561 (80.14%)**	387 (68.98%)	365 (94.32%)	0.94
	900	715 (79.44%)	425 (59.44%)	394 (92.71%)	0.94
	1000	795 (79.50%)	453 (56.98%)	412 (90.95%)	0.94
Uniform	100	**75 (75.00%)**	**63 (84.00%)**	**59 (93.65%)**	0.91
	300	224 (74.66%)	183 (81.69%)	170 (92.90%)	0.91
	500	364 (72.80%)	275 (75.54%)	252 (91.64%)	0.91
	700	502 (71.71%)	351 (69.92%)	312 (88.89%)	0.91
	900	617 (68.55%)	386 (62.56%)	342 (88.60%)	0.91
	1000	685 (68.50%)	401 (58.54%)	351 (87.53%)	0.91

Table 2 Evaluation results of different sampling methods on the QM9_13k subset

Sampling method	Sampling size	Validity ↑	Uniqueness ↑	Novelty ↑	Diversity ↑
Normal	100	72 (72.00%)	**71 (92.21%)**	66 (92.96%)	0.92
	300	232 (77.33%)	184 (79.31%)	**176 (95.65%)**	0.91
	500	378 (75.60%)	277 (73.28%)	257 (92.78%)	0.92
	700	542 (77.43%)	372 (68.63%)	346 (93.01%)	0.92
	900	**698 (77.56%)**	434 (62.18%)	409 (94.24%)	0.92
	1000	766 (76.60%)	444 (57.96%)	415 (93.47%)	0.92
Uniform	100	59 (59.00%)	**55 (93.22%)**	51 (92.73%)	0.92
	300	187 (62.33%)	148 (79.14%)	**145 (97.97%)**	0.92
	500	**329 (65.80%)**	230 (69.91%)	218 (94.78%)	0.92
	700	438 (62.57%)	287 (65.53%)	264 (91.99%)	0.91
	900	579 (64.33%)	351 (60.62%)	330 (94.02%)	0.92
	1000	656 (65.60%)	374 (57.01%)	354 (94.65%)	0.92

subset discussed earlier. In terms of effectiveness metrics, the model performs well on both sampling methods, with a high rate of valid molecule generation surpassing 62%. Additionally, the generated molecules exhibit a novelty of over 90%, indicating that the model has not only learned the correct molecular structures but also how to generate molecules that meet the requirements.

Considering both experiments, the overall performance of the model on the ZINC dataset is slightly better than that on the QM9 dataset, particularly in terms of generating valid molecules. The model demonstrates stronger learning capabilities for real molecules. It should be noted that in terms of uniqueness metrics, the model experiences a decrease as the sampling size increases for both datasets. The model performs better under normal noise sampling conditions compared to uniform noise sampling. This may be because normal noise can better cover the entire input space compared to uniform noise, thereby enhancing the model's exploratory and diversifying capabilities. It is more likely to guide the model to learn in different directions, helping to prevent the model from getting trapped in local optima.

Evaluation results under different training epochs For GAN models, stopping the training at an appropriate time can lead to better performance. We evaluated the model's performance using 50, 100, 150, and 200 training epochs. The training was conducted on two subsets: ZINC_25k and QM9_13k.

Table 3 presents the evaluation results of the model on the ZINC_25k subset for different training epochs. We can clearly observe that the model's performance is not optimal when the training epochs are less than 150. In terms of uniqueness, the model's effectiveness at 100 training epochs is even worse than that at 50 epochs. This is evident even when considering that the generated molecules have similar validity, indicating the instability of the model at this stage, which demonstrates the difficulty of training GAN models. However, as the training epochs increase to 150, the validity metric shows minimal fluctuations across the three sample sizes, and the decrease in uniqueness is smaller compared to that with fewer training epochs. Additionally, the rate of generating novel molecules does not decrease as a result. This suggests that the model performs exceptionally well and stably under the current training conditions. On the other hand, when we increase the training epochs to 200, the overall performance of the model starts to decline, resulting in a

Table 3 Evaluation results of different train epochs on the ZINC_25k subset

Epochs	Sampling size	Validity ↑	Uniqueness ↑	Novelty ↑	Diversity ↑
50	100	75 (75.00%)	**66 (88.00%)**	**62 (93.93%)**	0.92
	500	371 (74.20%)	243 (65.49%)	227 (93.41%)	0.92
	1000	**751 (75.10%)**	350 (46.60%)	315 (90.00%)	0.92
100	100	**77 (77.00%)**	**62 (80.51%)**	**59 (95.16%)**	0.92
	500	380 (76.00%)	226 (59.47%)	212 (93.80%)	0.92
	1000	763 (76.30%)	356 (46.65%)	334 (93.82%)	0.89
150	100	79 (79.00%)	**69 (87.34%)**	**67 (97.10%)**	0.92
	500	396 (79.20%)	294 (74.24%)	276 (93.88%)	0.94
	1000	**795 (79.50%)**	453 (56.98%)	412 (90.95%)	0.92
200	100	63 (63.00%)	**56 (88.887%)**	53 (94.64%)	0.93
	500	**326 (65.20%)**	239 (73.31%)	**228 (95.39%)**	0.93
	1000	651 (65.10%)	361 (55.15%)	341 (94.45%)	0.93

significant number of invalid molecules, which is evident in the validity metric. In conclusion, the evaluation results of this experiment indicate that the model's performance does not continuously improve with an increase in training epochs. The model achieves relatively optimal performance when trained for 150 epochs.

Table 4 presents the evaluation results of the model on the QM9_13k subset for different training epochs. Similarly, when the training epochs are set to 100, there is a slight improvement in the model's performance compared to 50 epochs, but it is not significant. However, as we further increase the training epochs to 150, there is a noticeable improvement in the model's performance. Although the improvement in validity is not substantial, with an overall increase of around 2%, there is a significant improvement in the uniqueness metric. When the sample size is 1000, the improvement reaches 10%. Moreover, the novelty metric of the model does not show a significant decline. On the other hand, when the training epochs reach 200, the same issue occurs, with a significant decline in the overall performance of the model. It even

Table 4 Evaluation results of different train epochs on the QM9_13k subset

Epochs	Sampling size	Validity ↑	Uniqueness ↑	Novelty ↑	Diversity ↑
50	100	**75 (75.00%)**	**65 (86.67%)**	62 (95.38%)	0.92
	500	348 (69.60%)	277 (58.33%)	257 (95.57%)	0.92
	1000	733 (73.30%)	327 (44.61%)	**314 (96.02%)**	0.92
100	100	74 (74.00%)	**65 (87.84%)**	**64 (98.46%)**	0.91
	500	**376 (75.20%)**	214 (56.91%)	205 (95.79%)	0.92
	1000	735 (73.50%)	346 (47.07%)	329 (95.09%)	0.92
150	100	**77 (77.00%)**	**66 (91.67%)**	59 (89.39%)	0.92
	500	378 (75.60%)	277 (73.28%)	257 (92.78%)	0.91
	1000	766 (76.60%)	444 (57.96%)	**415 (93.47%)**	0.92
200	100	56 (59.00%)	**55 (93.22%)**	51 (92.73%)	0.92
	500	**329 (65.80%)**	230 (69.91%)	**218 (94.78%)**	0.91
	1000	656 (65.60%)	374 (57.01%)	354 (94.65%)	0.92

performs the worst among all the training epochs. This demonstrates the instability of the model and highlights the importance of performance evaluation from another perspective.

Based on the recent two experiments, both on the ZINC_25k subset and the QM9_13k subset, it can be observed that the model achieves optimal performance at around 150 training epochs. However, as the training epochs continue to increase, a varying degree of performance decline occurs, especially in terms of the validity of generated molecules. This could be due to long training epochs leading to the issue of mode collapse, where the generator starts generating similar or even identical samples, lacking diversity and creativity. It could also be caused by improper adjustment of the learning rate, resulting in the model getting stuck in a local optimum. However, upon closer comparison between the two experiments, slight differences can still be observed. For instance, when the training epochs increase from 50 to 100 on the ZINC_25k subset, there is a slight improvement in the model's performance, whereas on the QM9_13k subset, there is no significant improvement until the training epochs reach 150. This may indicate that the model performs more stably on the ZINC_15k dataset, with performance gradually improving within a certain range of training epochs. This could be attributed to the fact that molecules in this dataset are longer, requiring more time for the model to learn their information.

Evaluation results under different data volumes The volume of the dataset has a certain impact on the model's ability to learn data distribution and molecular structures. Therefore, this study evaluates the performance of the model on homologous datasets with different data volumes, namely the ZINC dataset and its subsets, as well as the QM9 dataset and its subsets. Without further clarification, random sampling is used as the sampling method, and the training epochs are set to 150. In addition, for the purpose of distinguishing, we will refer to the original ZINC dataset as ZINC_250k and the QM9 dataset as QM9_130k.

Table 5 presents the evaluation results of the model on ZINC_25k and ZINC_250k datasets. It is evident that compared to training with a smaller dataset, the model exhibits a significant improvement in generating valid molecules when trained with a larger amount of training data. Even when the sampling data reaches its maximum, the validity metric remains above 82% and higher. This indicates that the model has learned more comprehensive structural information and sample distribution of molecules. Moreover, the occurrence

Table 5 The evaluation results of training data volumes on the ZINC dataset

Training data	Sampling size	Validity ↑	Uniqueness ↑	Novelty ↑	Diversity ↑
ZINC_25k	100	79 (79.00%)	**69 (87.34%)**	**67 (97.10%)**	0.94
	300	**238 (79.33%)**	193 (81.09%)	185 (95.85%)	0.94
	500	386 (77.20%)	294 (74.24%)	276 (93.88%)	0.94
	700	536 (76.57%)	387 (68.98%)	365 (94.32%)	0.94
	900	658 (73.11%)	425 (59.44%)	394 (92.71%)	0.94
	1000	723 (72.30%)	453 (56.98%)	412 (90.95%)	0.94
ZINC_250k	100	**85 (85.00%)**	**80 (94.11%)**	**36 (45.00%)**	0.93
	300	251 (83.67%)	216 (86.05%)	**98 (45.00%)**	0.93
	500	412 (82.40%)	335 (81.31%)	142 (42.39%)	0.93
	700	576 (82.28%)	416 (72.22%)	183 (43.99%)	0.93
	900	753 (83.67%)	501 (66.53%)	218 (43.51%)	0.93
	1000	823 (82.30%)	524 (63.67%)	226 (43.12%)	0.93

Table 6 The evaluation results of training data volumes on the QM9 dataset

Training data	Sampling size	Validity ↑	Uniqueness ↑	Novelty ↑	Diversity ↑
QM9_13k	100	77 (77.00%)	**66 (91.67%)**	59 (89.39%)	0.92
	300	232 (77.33%)	184 (79.31%)	**176 (95.65%)**	0.91
	500	378 (75.60%)	277 (73.28%)	257 (92.78%)	0.92
	700	542 (77.43%)	372 (68.63%)	346 (93.01%)	0.92
	900	**698 (77.56%)**	434 (62.18%)	409 (94.24%)	0.92
	1000	766 (76.60%)	444 (57.96%)	415 (93.47%)	0.92
QM9_130k	100	81 (81.00%)	**75 (92.59%)**	27 (36.00%)	0.92
	300	241 (80.33%)	192 (79.67%)	66 (34.38%)	0.92
	500	404 (80.80%)	297 (73.51%)	129 (43.43%)	0.92
	700	566 (80.86%)	389 (68.73%)	162 (41.65%)	0.92
	900	**749 (83.22%)**	475 (63.42%)	204 (42.95%)	0.91
	1000	796 (79.60%)	489 (61.43%)	**225 (44.98%)**	0.92

of mode collapse is not prominent on the ZINC_250k dataset. The uniqueness of generated molecules has not decreased due to the increase of valid molecules. In fact, it performs better than the results obtained from training on the ZINC_25k dataset under the same conditions. This suggests that with a sufficient amount of data, the model can learn more abundant information, leading to better performance. As for the novelty metric, a significant decrease is observed on the ZINC_250k dataset, which aligns with expectations. This is because as the dataset size increases, the molecular space becomes more constrained, resulting in a weaker exploration ability of the model.

Table 6 presents the evaluation results of the model on QM9_13k and QM9_130k datasets. From the data in the table, it can be observed that on the QM9_130k dataset, as the training data size increases, the model demonstrates better performance compared to training with a smaller dataset. In terms of the effectiveness metric, the model's performance improves by approximately 3% under the conditions of larger training data. Additionally, its performance on the uniqueness metric is slightly better than the training results on the smaller dataset. Similarly, both the model's novelty metric results and those on the ZINC dataset show a significant decrease, indicating that there is still room for improvement in the model's exploration ability.

Based on the two experiments conducted above, we can draw the conclusion that the performance of the model is related to the training data size. The larger the dataset, the more accurate the model learns the data distribution information and molecular structures, resulting in the generation of more valid molecules. Moreover, increasing the training data size also allows the model to reduce the generation of duplicate molecules. The decrease in diversity is the main area of model optimization. Encouraging the model to explore a more comprehensive chemical space and generate entirely novel molecules is a key problem to be addressed in the future.

Evaluation results of the important module In addition to the experiments mentioned above, we also conducted tests on the WGAN and reward network used in this paper. We replaced the Wasserstein distance with the JS divergence of the original GAN, and the evaluation results are presented in Table 7 and Fig. 3. The figure displays the uniqueness

Table 7 The evaluation results of distance measurement methods on the ZINC dataset

Measurement data	Method size	Validity ↑	Uniqueness ↑	Novelty ↑	Diversity ↑
Wasserstein distance	100	**85 (85.00%)**	**80 (94.11%)**	**36 (45.00%)**	0.93
	300	251 (83.67%)	216 (86.05%)	**98 (45.00%)**	0.93
	500	412 (82.40%)	335 (81.31%)	142 (42.39%)	0.93
	700	576 (82.28%)	416 (72.22%)	183 (43.99%)	0.93
	900	753 (83.67%)	501 (66.53%)	218 (43.51%)	0.93
	1000	823 (82.30%)	524 (63.67%)	226 (43.12%)	0.93
JS divergence	100	**87 (87.00%)**	**68 (78.16%)**	32 (47.06%)	0.80
	300	254 (84.66%)	169 (66.53%)	78 (46.15%)	0.75
	500	425 (85.00%)	231 (54.35%)	**115 (49.78%)**	0.74
	700	588 (84.00%)	284 (48.30%)	132 (46.48%)	0.73
	900	766 (85.11%)	335 (43.37%)	158 (47.16%)	0.75
	1000	843 (84.30%)	354 (42.00%)	164 (46.32%)	0.74

and novelty metrics of the models generated on the ZINC dataset under two different conditions. Since we found that the validity metric was minimally affected in the experiment, it was not shown. From the figure, it can be seen that the uniqueness metric of molecules is significantly affected by WGAN, which is explainable. This is because WGAN is certain to play a positive role in mitigating mode collapse. As for the novelty metric, although the use of JS divergence shows higher maturity, this is also based on proportions; in terms of quantity, using the Wasserstein distance is still better than JS divergence.

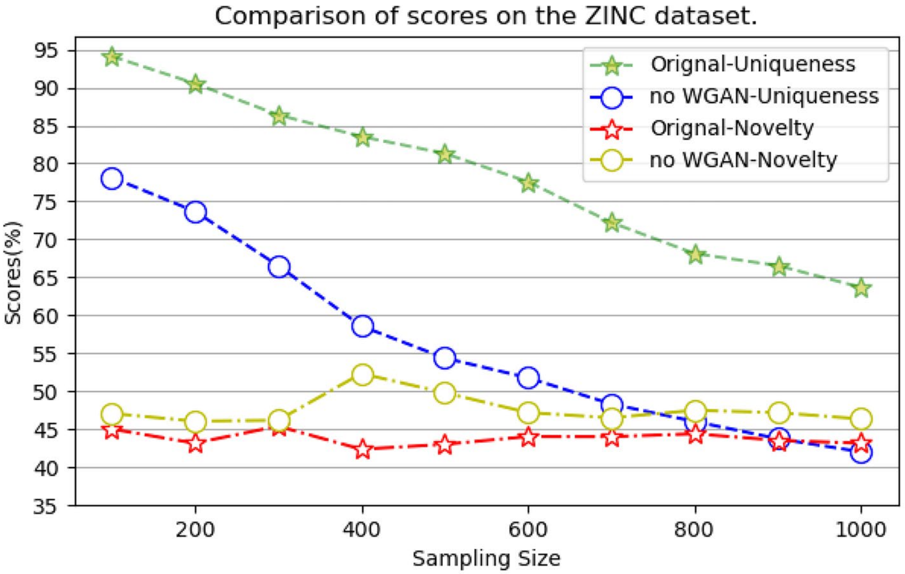


Fig. 3 Comparison of evaluation metrics with and without using WGAN on the ZINC dataset. The lines marked with asterisks represent the Wasserstein distance, while those marked with circles represent the JS distance

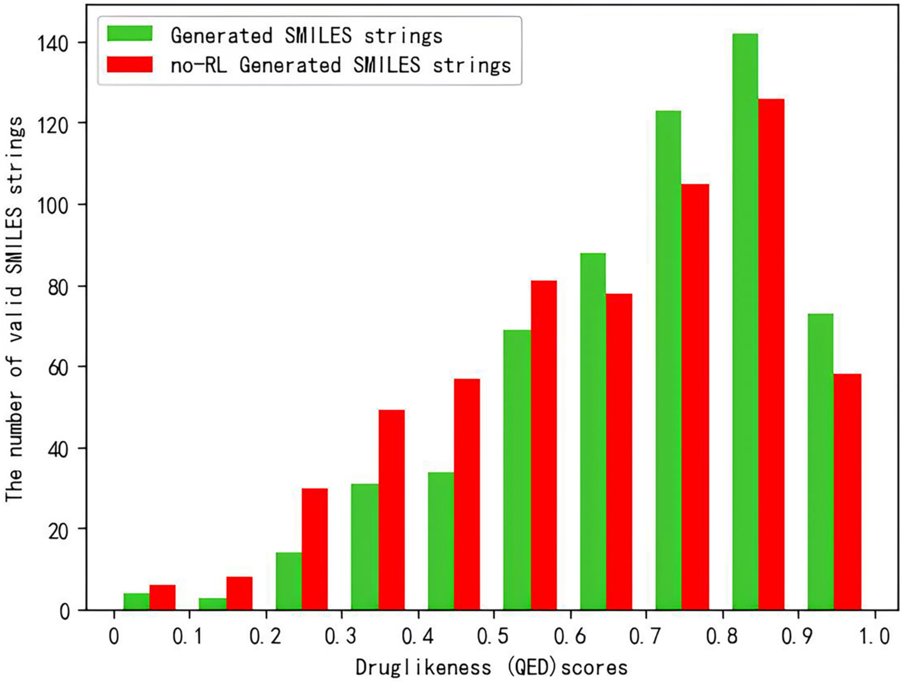


Fig. 4 Distribution of QED scores between generated samples with the standard model and the model without using the reward network

For the evaluation of the reward network, we only conducted experiments on molecular properties that have a significant impact. Figures 4, 5 and 6 respectively show the distributions of QED, LogP, and SA scores of molecules generated by the standard model and the model without using the reward network. The green bars in the figures represent the number of molecules generated by the standard model, while the corresponding red bars represent the number of molecules generated without using the reward network. From the three figures, it is evident that the properties of molecules generated by the model are affected when the reward network part is removed. Overall, the drug-likeness and overall score of the molecules show a significant decrease, and the water solubility also weakens.

Figure 7 shows the comparison of the distribution of QED scores between the molecules generated by the model on the ZINC dataset and the molecules in the original training dataset. Similarly, Figs. 8 and 9 show the comparison of LogP scores and SA scores respectively. The generated data was sampled using the normal method, trained for 150 epochs, and a total of 580 molecules were sampled. From the figures, it can be observed intuitively how well the model has learned from the original data. For all three evaluation metrics, the model is capable of generating samples that closely resemble the original data. Figures 10, 11 and 12 show the comparison of the distribution of QED scores, LogP scores and SA scores between the molecules generated by the model on the ZINC dataset and the molecules in the original training dataset respectively. Figure 13 shows the generated and real molecules on the QM9 dataset, respectively. The generated molecular graph exhibits

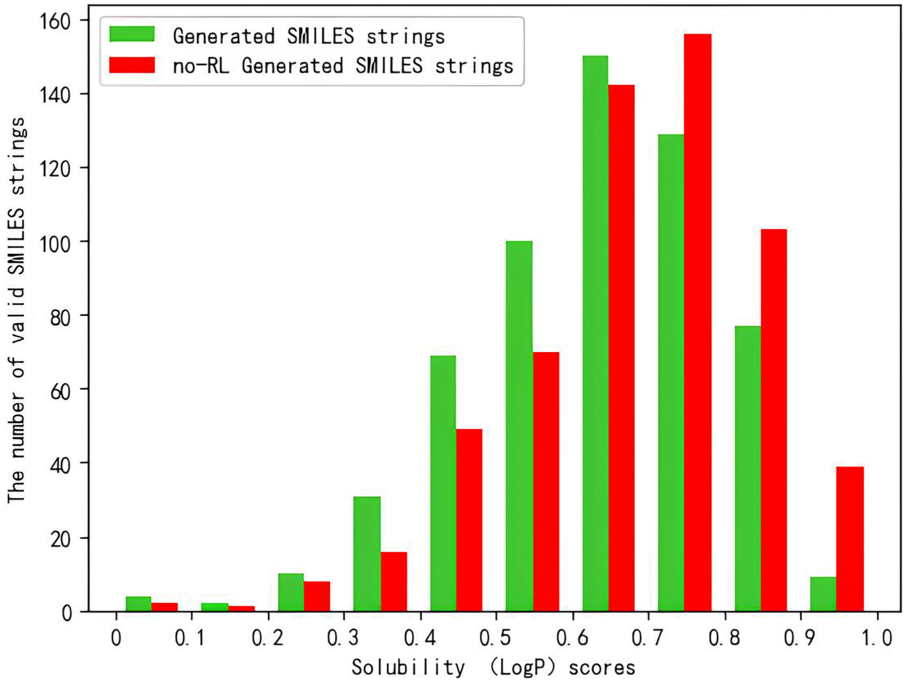


Fig. 5 Distribution of LogP scores between generated samples with the standard model and the model without using the reward network

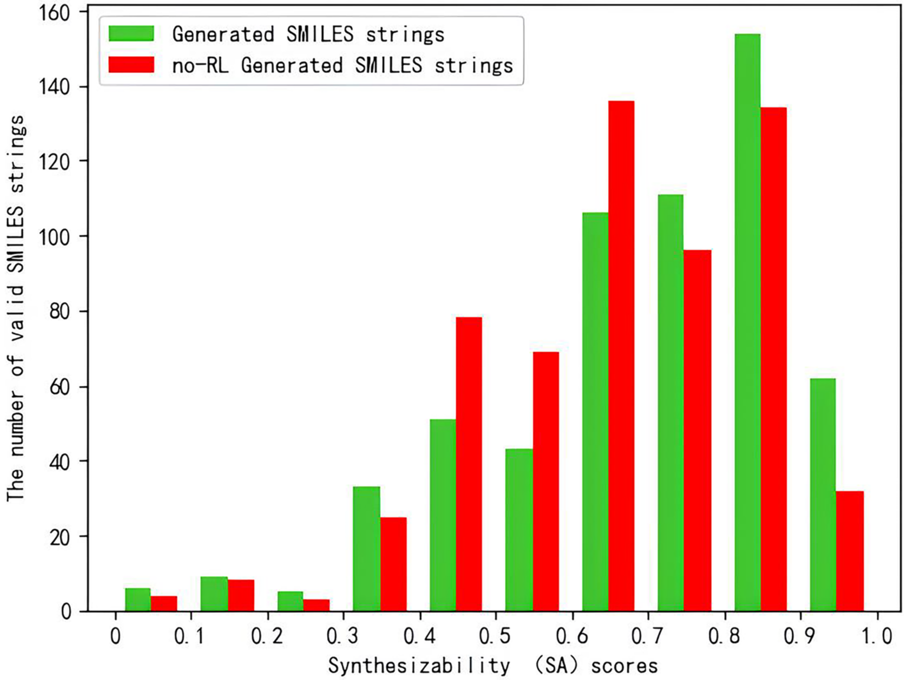


Fig. 6 Distribution of SA scores between generated samples with the standard model and the model without using the reward network

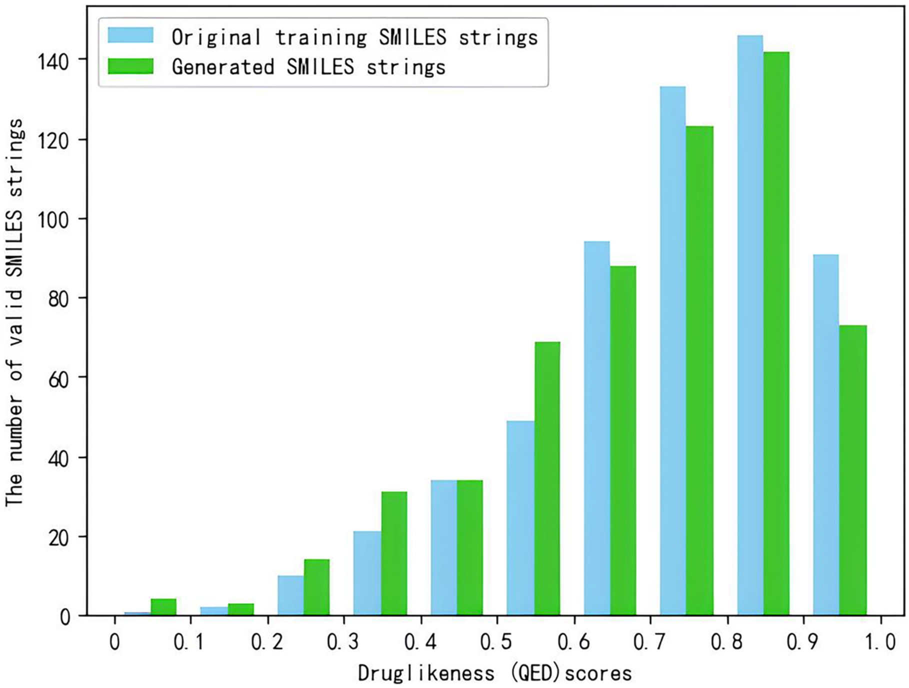


Fig. 7 Distributions of QED scores between the generated samples and samples the ZINC dataset

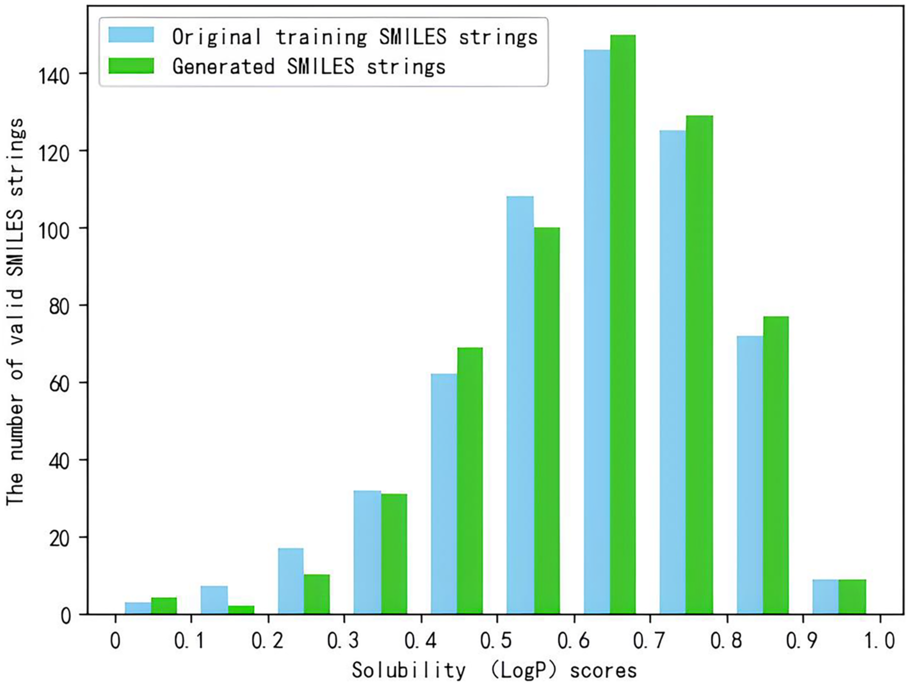


Fig. 8 Distributions of LogP scores between the generated samples and samples in the ZINC dataset

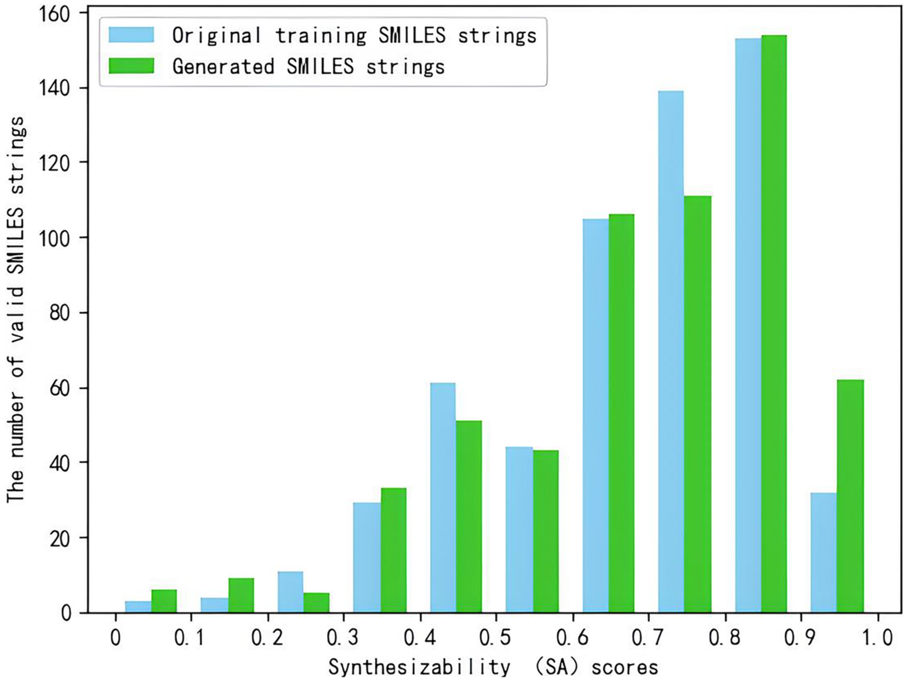


Fig. 9 Distributions of SA scores between the generated samples and samples in the ZINC dataset

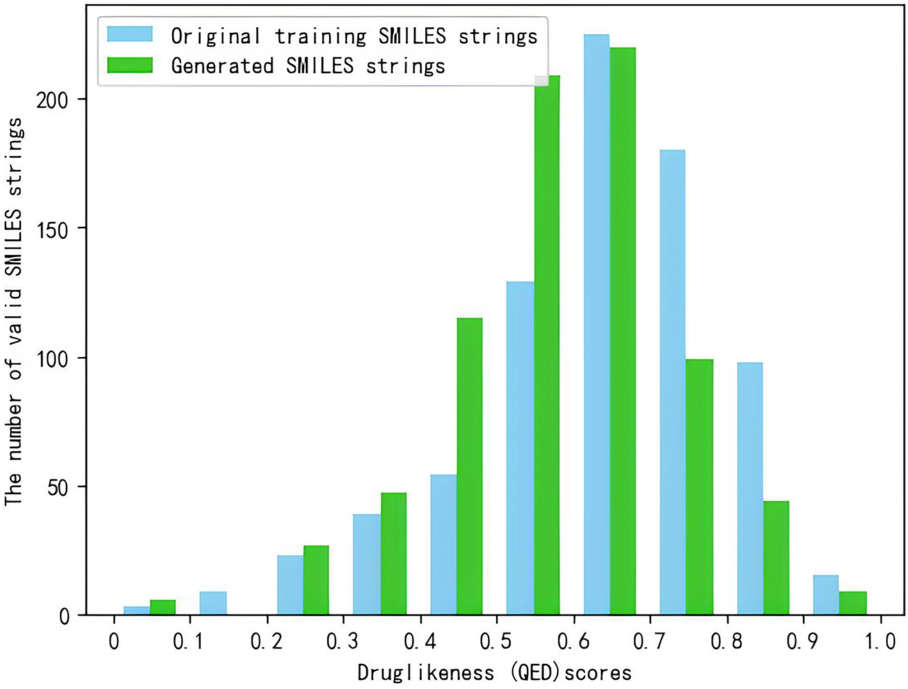


Fig. 10 Distributions of QED scores between the generated samples and samples in the QM9 dataset

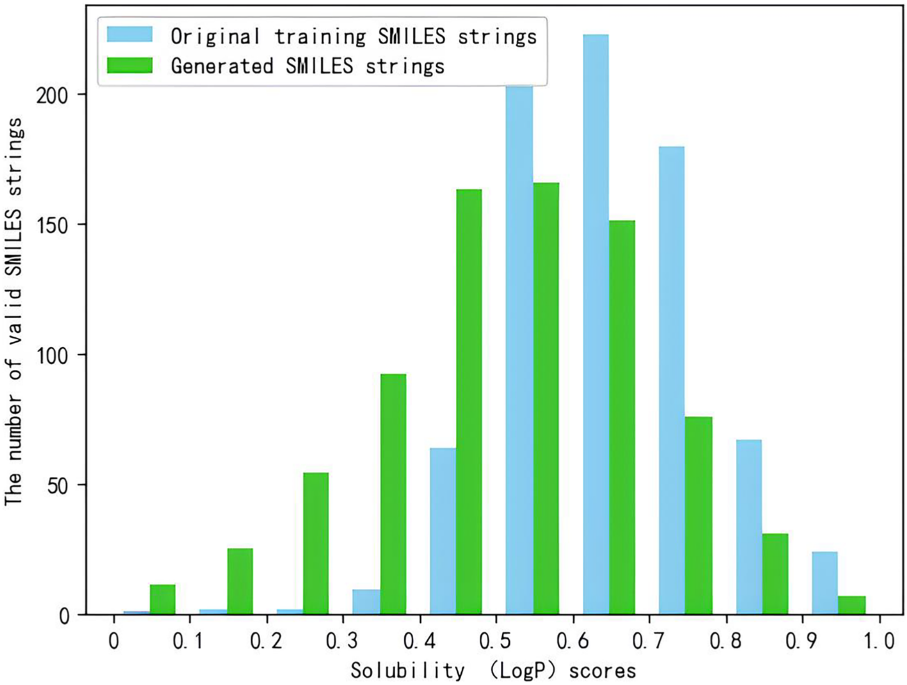


Fig. 11 Distributions of LogP scores between the generated samples and samples in the QM9 dataset

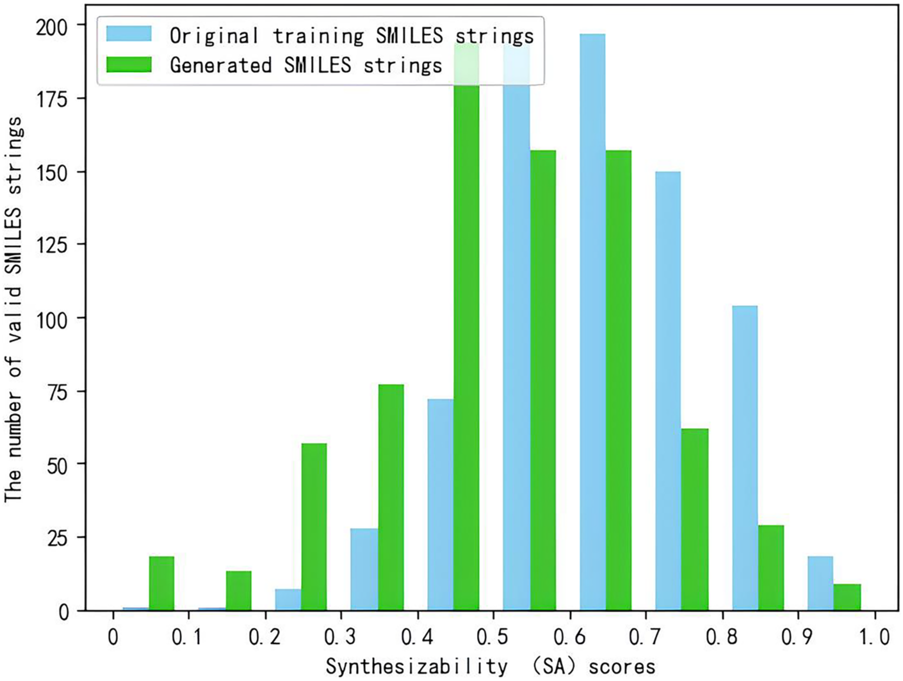
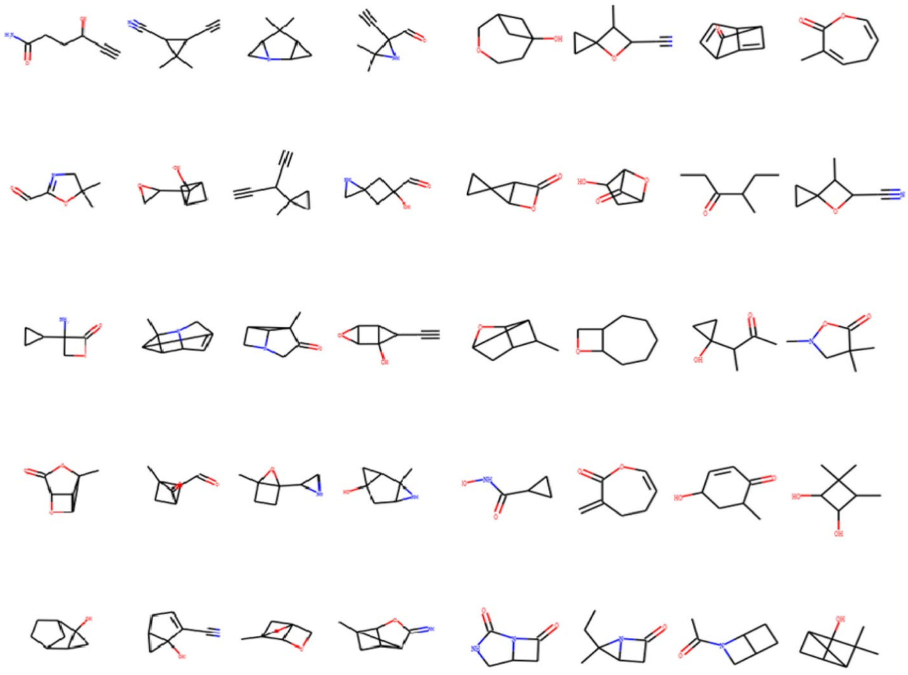


Fig. 12 Distributions of SA scores between the generated samples and samples in the QM9 dataset



(a) Molecules of the QM9 dataset.

(b) Newly generated molecules.

Fig. 13 20 real and generated molecules from the QM9 dataset

some similarity with real data, but the generated molecules by the RL-based discrete GAN tend to have a high diversity than the original training dataset.

6 Conclusion

This study evaluates the use of RL-based discrete GAN models for molecular generation tasks on two different datasets. We also conduct a detailed performance evaluation of the model by varying the sampling method, number of training epochs, and training data volumes. Through analysis of statistical and attribute values in the experimental results, it is observed that, under the condition of other variables being constant, the model achieves better performance when using the normal sampling method. When evaluating the model by controlling other variables and varying the number of training epochs, the best performance is achieved when the number of epochs is set to 150 for both datasets. Performance deteriorates beyond this value. Increasing the training data volumes improves the model's performance but also leads to a notable decrease in the novelty of generated molecules, indicating a key area for model optimization.

When comparing the two datasets, a common observation is that the unique valid molecule rate decreases as the sampling quantity increases, indicating the persistence of mode collapse in the model. However, there are differences between the datasets. The overall performance of the model on the ZINC dataset is superior to that on the QM9 dataset. This can be attributed to the ZINC dataset containing more realistic and diverse molecules with larger molecular weights, allowing the model to learn more about the distribution and molecular structure information of the original data.

Based on the above conclusions, the evaluated model in this paper still suffers from mode collapse. Since molecular data is typically sparse and high-dimensional, data augmentation and preprocessing techniques can be employed to improve the training effectiveness of GANs. For example, data augmentation techniques can be used to increase the size of the training set or dimensionality reduction methods can be applied to reduce the dimensionality of the data, thereby enhancing the learning efficiency of GANs. Additionally, conditional GANs can be utilized to introduce conditional information, such as specific chemical properties or structural constraints. By controlling this conditional information, we can guide the generator to generate more diverse molecular structures and avoid the issue of mode collapse.

Acknowledgements We would like to acknowledge the financial support provided by two grants, the JSPS KAKENHI Grant Number JP22K21285, Japan and the International Research Fellow of Japan Society for the Promotion of Science (Postdoctoral Fellowships for Research in Japan [Standard]), which made this research possible.

Author contributions Jinli Zhang: methodology, writing, and supervision. Zhenbo Wang: software, experiments, analysis, and writing. Zongli Jiang: supervision and proofreading. Man Wu: supervision and proofreading. Chen Li: conceptualization of this study, methodology, writing, and supervision. Yoshihiro Yamanishi: supervision and proofreading.

Funding This work was supported by the JSPS KAKENHI Grant Number JP22K21285, Japan and the International Research Fellow of Japan Society for the Promotion of Science (Postdoctoral Fellowships for Research in Japan [Standard]).

Availability of data and material The authors confirm that the data supporting the findings of this study are available within the article.

Code availability Source code is available on request.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Competing interests The authors declare no competing interests.

References

An, Z., Tan, Y., Zhang, J., Jiang, Z., & Li, C. (2023). A session recommendation model based on heterogeneous graph neural network. *International Conference on Knowledge Science, Engineering and Management* (pp. 160–171). Springer.

Bagal, V., Aggarwal, R., Vinod, P., & Priyakumar, U. D. (2021). MolGPT: Molecular generation using a transformer-decoder model. *Journal of Chemical Information and Modeling, 62*(9), 2064–2076.

Bao, F., Zhao, M., Hao, Z., Li, P., Li, C., & Zhu, J. (2022). Equivariant energy-guided SDE for inverse molecular design. Preprint retrieved from http://arxiv.org/abs/2209.15408

Bickerton, G. R., Paolini, G. V., Besnard, J., Muresan, S., & Hopkins, A. L. (2012). Quantifying the chemical beauty of drugs. *Nature chemistry, 4*(2), 90–98.

Chen, L., Zheng, J., Okamura, H., & Dohi, T. (2022). Software reliability prediction through encoder-decoder recurrent neural networks. *International Journal of Mathematical, Engineering and Management Sciences, 7*(3), 325.

Comer, J., & Tam, K. (2001). Lipophilicity profiles: theory and measurement. *Pharmacokinetic Optimization in Drug Research: Biological, Physicochemical, and Computational Strategies*, 275–304.

De Cao, N., & Kipf, T. (2018). MolGAN: An implicit generative model for small molecular graphs. Preprint retrieved from http://arxiv.org/abs/1805.11973

Ertl, P., & Schuffenhauer, A. (2009). Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of Cheminformatics, 1*, 1–11.

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., & Aspuru-Guzik, A. (2018). Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science, 4*(2), 268–276.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM, 63*(11), 139–144.

Guimaraes, G.L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C., & Aspuru-Guzik, A. (2017). Objective-reinforced generative adversarial networks (organ) for sequence generation models. Preprint retrieved from http://arxiv.org/abs/1705.10843

Hoogeboom, E., Satorras, V. G., Vignac, C., & Welling, M. (2022). Equivariant diffusion for molecule generation in 3D. *International Conference on Machine Learning* (pp. 8867–8887). PMLR.

Huang, L., Zhang, H., Zhang, T., & Wong, K.-C. (2023). MDM: Molecular diffusion model for 3D molecule generation. *Proceedings of the AAAI Conference on Artificial Intelligence, 37*, 5105–5112.

Irwin, J. J., Sterling, T., Mysinger, M. M., Bolstad, E. S., & Coleman, R. G. (2012). ZINC: A free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling, 52*(7), 1757–1768.

Jiang, Z., Xu, J., Zhang, J., Ma, F., & Li, J. (2022). Dual memory network for medical dialogue generation. In *2022 IEEE 24th International Conference on High Performance Computing & Communications; 8th International Conference on Data Science & Systems; 20th International Conference on Smart City; 8th International Conference on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)* (pp. 110–117). IEEE.

Jin, W., Barzilay, R., & Jaakkola, T. (2018). Junction tree variational autoencoder for molecular graph generation. *International Conference on Machine Learning* (pp. 2323–2332). PMLR.

Kadurin, A., Nikolenko, S., Khrabrov, K., Aliper, A., & Zhavoronkov, A. (2017). druGAN: An advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Molecular Pharmaceutics, 14*(9), 3098–3104.

Kim, S., Thiessen, P. A., Bolton, E. E., Chen, J., Fu, G., Gindulyte, A., Han, L., He, J., He, S., Shoemaker, B. A., et al. (2016). Pubchem substance and compound databases. *Nucleic Acids Research, 44*(D1), 1202–1213.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. Preprint retrieved from http://arxiv.org/abs/1312.6114

Kusner, M. J., Paige, B., & Hernández-Lobato, J. M. (2017). Grammar variational autoencoder. *International Conference on Machine Learning* (pp. 1945–1954). PMLR.

Li, C., Cao, Y., Zhu, Y., Cheng, D., Li, C., & Morimoto, Y. (2023). Ripple knowledge graph convolutional networks for recommendation systems. Preprint retrieved from http://arxiv.org/abs/2305.01147

Li, C., He, M., Qaosar, M., Ahmed, S., & Morimoto, Y. (2018). Capturing temporal dynamics of users' preferences from purchase history big data for recommendation system. *2018 IEEE International Conference on Big Data (Big Data)* (pp. 5372–5374). IEEE.

Li, C., & Yamanishi, Y. (2023). SpotGAN: A reverse-transformer GAN generates scaffold-constrained molecules with property optimization. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 323–338). Springer.

Li, C., Yamanaka, C., Kaitoh, K., & Yamanishi, Y. (2021). Transformer-based objective-reinforced generative adversarial network to generate desired molecules. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22* (pp. 3884–3890)

Li, Y., Tarlow, D., Brockschmidt, M., & Zemel, R. (2015). Gated graph sequence neural networks. Preprint retrieved from http://arxiv.org/abs/1511.05493

Lillicrap, T.P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015). Continuous control with deep reinforcement learning. Preprint retrieved from http://arxiv.org/abs/1509.02971

Maziarka, Ł, Pocha, A., Kaczmarczyk, J., Rataj, K., Danel, T., & Warchoł, M. (2020). Mol-CycleGAN: A generative model for molecular optimization. *Journal of Cheminformatics, 12*(1), 1–18.

Morehead, A., & Cheng, J. (2023). Geometry-complete diffusion for 3D molecule generation. Preprint retrieved from http://arxiv.org/abs/2302.04313

Ramakrishnan, R., Dral, P. O., Rupp, M., & Von Lilienfeld, O. A. (2014). Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data, 1*(1), 1–7.

Rifaioglu, A. S., Cetin Atalay, R., Cansen Kahraman, D., Doğan, T., Martin, M., & Atalay, V. (2021). MDeePred: Novel multi-channel protein featurization for deep learning-based binding affinity prediction in drug discovery. *Bioinformatics, 37*(5), 693–704.

Rogers, D. J., & Tanimoto, T. T. (1960). A computer program for classifying plants. *Science, 132*(3434), 1115–1118.

Samanta, B., De, A., Ganguly, N., & Gomez-Rodriguez, M. (2018). Designing random graph models using variational autoencoders with applications to chemical design. Preprint retrieved from http://arxiv.org/abs/1802.05283

Sanchez-Lengeling, B., Outeiral, C., Guimaraes, G. L., & Aspuru-Guzik, A. (2017). Optimizing distributions over molecular space. An objective-reinforced generative adversarial network for inverse-design chemistry (organic). ChemRxiv.

Sarpong, D., Boakye, D., Ofosu, G., & Botchie, D. (2023). The three pointers of research and development (r &d) for growth-boosting sustainable innovation system. *Technovation, 122*, 102581.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., & Welling, M. (2018). Modeling relational data with graph convolutional networks. *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15* (pp. 593–607). Springer.

Shi, C., Xu, M., Zhu, Z., Zhang, W., Zhang, M., & Tang, J. (2020) GraphAF: A flow-based autoregressive model for molecular graph generation. Preprint retrieved from http://arxiv.org/abs/2001.09382

Simonovsky, M., & Komodakis, N. (2018). GraphVAE: Towards generation of small graphs using variational autoencoders. *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4–7, 2018, Proceedings, Part I 27* (pp. 412–422). Springer.

Song, T., Ren, Y., Wang, S., Han, P., Wang, L., Li, X., & Rodriguez-Patón, A. (2023). DNMG: Deep molecular generative model by fusion of 3D information for de novo drug design. *Methods, 211*, 10–22.

Walker, A. R. (1998). Epidemiology and health implications of obesity, with special reference to African populations. *Ecology of Food and Nutrition, 37*(1), 21–55.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement Learning*, 5–32.

Xu, M., Powers, A. S., Dror, R. O., Ermon, S., & Leskovec, J. (2023). Geometric latent diffusion models for 3D molecule generation. *International Conference on Machine Learning* (pp. 38592–38610). PMLR.

You, J., Ying, R., Ren, X., Hamilton, W., & Leskovec, J. (2018). GraphRNN: Generating realistic graphs with deep auto-regressive models. *International Conference on Machine Learning* (pp. 5708–5717). PMLR.

Yu, L., Zhang, W., Wang, J., & Yu, Y. (2017). SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence* (p. 31)

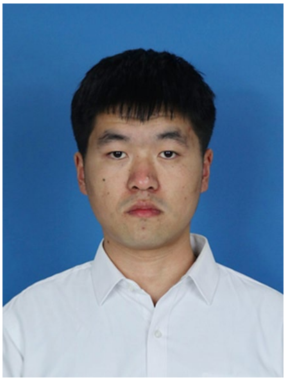
Zhao, B., Jiang, Z., Zhang, J., Ma, F., & Li, J. (2022). Medical dialogue generation via extracting heterogenous information. *In 2022 IEEE 24th International Conference on High Performance Computing & Communications; 8th International Conference on Data Science & Systems; 20th International Conference on Smart City; 8th International Conference on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)* (pp. 201-IEEE). 194.

Zhang, X., Li, C., & Morimoto, Y. (2019). A multi-factor approach for stock price prediction by using recurrent neural networks. *Bulletin of Networking, Computing, Systems, and Software, 8*(1), 9–13.

Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *In Proceedings of the IEEE International Conference on Computer Vision* (pp. 2223–2232)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Jinli ZHANG received her PhD Degree from Beijing University of Technology, China, in 2019. She undertook joint doctoral training at Drexel University in the United States in 2018, 2019, and 2020. Since 2020, she has been working as a faculty member at the School of Computer Science, Faculty of Informatics, Beijing University of Technology, China. Her research interests include deep learning and bioinformatics mining.



Zhenbo WANG received the B.S degree in Internet of Things Engineering from Shandong University of Science and Technology, China, in 2021. He is currently working toward the M.S degree in Computer Science and Technology with the Faculty of Information Technology, Beijing University of Technology, China. His research interests include machine learning and deep learning.

Zongli JIANG is a professor at the School of Computer Science, Faculty of Informatics, Beijing University of Technology. He has served as Vice Chairman of the Teaching Guidance Committee for Computer Majors in Higher Education Institutions under the Ministry of Education, as well as positions such as President and Vice President of the National Association of Computer Education in Higher Education Institutions. He has been awarded various honors, including the National Teaching Master Award and the CCF Outstanding Education Award. He has edited four national planning textbooks and one national quality textbook. His research interests include artificial intelligence and machine learning.

Man WU received her Ph.D. degree from Nara Institute of Science and Technology, Japan, in 2022. Since 2022, she has been a research fellow at the Faculty of Science and Technology, Keio University, Japan. Her research interests include deep learning and accelerators.

Chen LI received his PhD in engineering from Hiroshima University, Japan, in 2019. He served as a research fellow at the Graduate School of Advanced Science and Engineering, Hiroshima University, from 2019 to 2021, and subsequently at the Department of Bioscience and Bioinformatics, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology, Japan, from 2021 to 2023. Since 2023, he has been a JSPS research fellow at the Graduate School of Informatics, Nagoya University, Japan. His research interests include deep learning and data mining.

Yoshihiro YAMANISHI received the PhD degree from Kyoto University, in 2005. He was a post-doctoral fellow at the Ecole des Mines de Paris from 2005 to 2006. He was assistant professor at Kyoto University from 2006 to 2007. He was a permanent researcher at Mines ParisTech and Curie Institute from 2008 to 2012. Since 2012, he was an associate professor (principal investigator) at the Medical Institute of Bioregulation, Kyushu University, Japan. From 2018 to 2023, he was a full professor with the Department of Bioscience and Bioinformatics, Kyushu Institute of Technology, Japan. Since 2023, he has been a full professor with the Department of Complex Systems Science, Graduate School of Informatics, Nagoya University, Japan. His research interest is in statistical machine learning methods for bioinformatics, chemoinformatics, and genomic drug discovery.