Transformer-Based Objective-Reinforced Generative Adversarial Network to Generate Desired Molecules

Chen Li, Chikashige Yamanaka, Kazuma Kaitoh and Yoshihiro Yamanishi
Department of Bioscience and Bioinformatics, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology, Iizuka, Japan
li260@bio.kyutech.ac.jp, yamanaka.chikashige215@mail.kyutech.jp, {kaito168, yamani}@bio.kyutech.ac.jp

Abstract

Deep generative models of sequence-structure data have attracted widespread attention in drug discovery. However, such models cannot fully extract the semantic features of molecules from sequential representations. Moreover, mode collapse reduces the diversity of the generated molecules. This paper proposes a transformer-based objective-reinforced generative adversarial network (TransORGAN) to generate molecules. TransORGAN leverages a transformer architecture as a generator and uses a stochastic policy gradient for reinforcement learning to generate plausible molecules with rich semantic features. The discriminator grants rewards that guide the policy update of the generator, while an objective-reinforced penalty encourages the generation of diverse molecules. Experiments were performed using the ZINC chemical dataset, and the results demonstrated the usefulness of TransORGAN in terms of uniqueness, novelty, and diversity of the generated molecules.

1 Introduction

The goal of drug discovery is to produce new compounds for treating diseases. However, the path to drug approval has become increasingly complicated and expensive. Over the past decade, the development cost of a new prescription medication that gains market approval has risen by 145% to 2.6 billion USD, and the average development time is 10 years. To accelerate this process, medical scientists have begun using artificial intelligence (AI) and deep learning to perform tasks that once depended on human intelligence. By using advanced techniques, medical researchers at the forefront of drug development can gain timely and actionable insights from stacks of unstructured data.

Various deep generative models for de novo molecular generation have been explored recently, including variational auto-encoders (VAEs) [Kusner et al., 2017; Dai et al., 2018; Jin et al., 2018] and generative adversarial networks (GANs) [Guimaraes et al., 2017; De Cao and Kipf, 2018]. A GAN [Goodfellow et al., 2014] is an unsupervised deep learning approach for generative modeling and has two main components: a generator and discriminator. The generator attempts to generate realistic fake data, and the discriminator aims to distinguish synthetic data from the original data. For molecular generation, the simplified molecular-input line-entry system (SMILES) is commonly used to represent molecules as string-based sequences derived from molecular graphs [Weininger, 1988]. For example, the molecular structure “Ic1cnccn1” begins with the iodine atom “I” followed by the ring structure “c1cnccn1.” Therefore, using a GAN with SMILES to generate new molecules seems to be a reasonable approach. Unfortunately, there are two problems with sequence generation by a GAN. First, a GAN reliably generates real-valued continuous data, but it is unsuitable for indirect sequence generation of discrete tokens such as SMILES strings. Second, a GAN can only give the score/loss of an entire string. Balancing the current score of a partially generated subsequence with the future score of the complete sequence is a nontrivial task.

SeqGAN [Yu et al., 2017] and objective-reinforced GAN (ORGAN) [Guimaraes et al., 2017] were developed to address the above problems. SeqGAN uses a reinforcement learning (RL) approach [Sutton et al., 2000] to facilitate the generation of discrete data. ORGAN is a SeqGAN-based model that accounts for chemical properties such as the solubility and drug-likeness of generated molecules. Both SeqGAN and ORGAN employ recurrent neural networks (RNNs), especially long short-term memory (LSTM), as the generator. However, RNNs have certain limitations when generating molecules with SMILES representations. First, RNNs have difficulty with designing a molecule with complex rings [Artis-Pous et al., 2019]. In general, highly cyclic molecules have long sequence representations and a stricter syntax than acyclic molecules. Slight changes in syntax may result in the generation of invalid molecules or molecules with very different chemical properties from those intended. Second, RNNs cannot work on GPU versions because the current iteration must be computed after the previous time step, which is not conducive to exploring the near-infinite chemical spaces of big data.
A transformer is a self-attention-based neural network architecture that provides state-of-the-art performance for neural machine translation tasks [Vaswani et al., 2017]. A transformer can effectively solve the shortcomings of traditional RNNs in iterative calculations and uses the self-attention mechanism to capture useful syntax information. However, transformer-based GANs face several problems with molecular generation. First, because the input and output are the same, a transformer is likely to remember only that the tokens at the corresponding positions are the same, which results in insufficient training. In addition, simply applying RL approaches, such as a policy gradient to train the transformer may lead to instability.

To overcome these drawbacks, we propose a transformer-based ORGAN (TransORGAN) model. The generator is a transformer that outputs likely molecules as SMILES strings. In addition, the same molecule can have various SMILES representations, called variant SMILES, which are used by the model to sufficiently learn the syntactic and semantic rules of highly cyclic molecules. The discriminator evaluates the generated data and feedbacks the rewards that guide the generator according to the RL method. The main contributions are summarized below:

- A transformer is employed as the generator. A transformer architecture can be parallelized, and the self-attention mechanism can capture the rich semantic information of molecules represented by SMILES strings.
- Variant SMILES allows the transformer to sufficiently learn the syntax rules with different strings and increase the diversity of generated molecules.
- The teacher-forcing improves the stability of the policy gradient when it is guiding the generation of molecules with the desired chemical properties during training.

2 Related Works

Deep generative models have attracted widespread attention, because they can learn coherent latent representations of continuous data such as image and video data. However, generating molecular structures and other discrete data remains a challenging task. Molecules generated from discrete data are often invalid. Two kinds of VAEs have been proposed for generating molecules: Character-VAE and Grammar-VAE [Kusner et al., 2017]. Character-VAE can generate molecules from characters without constraints, while Grammar-VAE applies a parse tree to constrain the grammar of SMILES. Grammar-VAE can generate syntactically valid molecules, but it cannot generate semantics. A syntax-direct VAE (SD-VAE) model was proposed to generate syntactic and semantic valid molecules [Dai et al., 2018]. A junction tree-based VAE (JT-VAE) can generate molecular graphs by using a graph message-passing network [Jin et al., 2018]. A VAE and its variants attempt to extract features of the encoder into a fixed-size latent space, which limits the ability of decoders to decode the latent features. Graph-AF is a flow-based autoregressive neural network that can generate molecules from molecular graphs [Shi et al., 2019]. Although graphical representations of molecules contain rich semantic and syntactic information, graph-based models are generally more complicated than SMILES strings.

A GAN alternately conducts adversarial training to improve the generator’s ability to generate data and the discriminator’s ability to distinguish between synthetic and real data. Although GANs are not limited to a fixed-size latent space, they have difficulties with generating discrete data sequences. SeqGAN considers the sequence generation procedure as a sequential decision-making process [Yu et al., 2017]. The generator is an agent of RL, the state comprises the tokens generated thus far, and the action is the next token. The discriminator evaluates the sequence and provides feedback to guide the learning of the generator. However, GANs cannot back propagate gradients to the generator when the output is discrete. In SeqGAN, the generator is a stochastic parametrized policy. The policy gradients approximate state-action values using a Monte Carlo search process. To consider desired chemical properties, ORGAN extended the training process of SeqGAN to include not only the discriminator reward but also domain-specific objectives [Guimaraes et al., 2017]. Specifically, ORGAN weights the molecular objectives by a factor $\lambda$. When $\lambda = 0$, ORGAN becomes a Naïve RL that aims only to generate valid molecules. Alternatively, when $\lambda = 1$, ORGAN becomes a simple SeqGAN that does not consider the molecular properties. MolGAN is an implicit and likelihood-free graph-structure-based GAN for generating small molecular structures [De Cao and Kipf, 2018]. However, although MolGAN improved the validity of the generated molecules, mode collapse reduced the uniqueness of the generated molecules to less than 3.2%.

In most of the above works, the GAN generator was an RNN, which cannot sufficiently extract the latent features from a long sequence with rich semantic information such as SMILES strings. A transformer [Vaswani et al., 2017] can capture semantic features in sentences by using a self-attention mechanism alone without needing an RNN or convolution. Molecular generation can be regarded as a special type of natural language processing that treats SMILES strings as sentences. In this case, a transformer can be applied to generate molecules. A transformer-based neural network has been proposed that uses SMILES data to generate molecules with a desirable balance of properties [He et al., 2021]. A novel architecture based on a transformer and spatial graph convolutional networks (GNNs) has also been proposed that enhances the attention mechanism of a transformer based on the interatomic distances and graph structures of molecules [Danel et al., 2020]. However, few works have combined a transformer with a GAN to improve the quality of generation. In addition, mode collapse during the adversarial training process is a common cause of low uniqueness [De Cao and Kipf, 2018]. Our proposed TransORGAN, which enhances the semantic feature extraction for molecular generation and imposes additional objective-reinforced penalties with RL to alleviate mode collapse.

3 TransORGAN

Figure 1 shows the architecture of TransORGAN, which consists of two main components: a generator and discrimina-
The generator and discriminator act as two players in a minimax game. The generator tries to generate high-quality molecules represented as SMILES strings under the guidance of the discriminator through RL. Meanwhile, the discriminator tries to distinguish the generated fake data from real molecules and to avoid being fooled by the generator. Note that the generator and discriminator are trained in alternation. Formally, let $G_{\theta}$ and $D_{\phi}$ represent the generator and discriminator, respectively, where $\theta$ and $\phi$ are parameters. The minimax game can then be expressed as follows:

$$\min_{\theta} \max_{\phi} V(G_{\theta}, D_{\phi}) = \mathbb{E}_{x \sim p_{data}(x)}[\log D_{\phi}(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_{\phi}(G_{\theta}(z)))]$$

### 3.1 Variant SMILES

In general, the input of a transformer differs from the output, which facilitates model learning. However, TransORGAN uses only SMILES data, which are not paired. If the same SMILES string is used as the input and output, the generator of TransORGAN can only learn a one-to-one correspondence between the input and output, so the training is insufficient. Insufficient learning by the generator may lead to mode collapse. We considered assigning different atomic correspondence between the input and output, so the training is insufficient. Insufficient learning by the generator may lead to mode collapse. We considered assigning different atomic numbers as training labels to each SMILES string. Fortunately, a molecule can be represented by various strings [Jerrum, 2017]. For example, “CC(=O)Oc1ccccc1C(=O)O” and “c1ccc(c(c1)C(O)=O)OC(O)=O” are different strings representing the same molecule. Figure A.1 and Algorithm A.1 in the Appendix describe how to produce variants for SMILES.

### 3.2 Generator Training Process

A transformer can be represented by an encoder-decoder paradigm and trained in an end-to-end fashion. An encoder uses self-attention to extract features of the input sequence in parallel. These features are used as the input of the decoder. In the decoder, sequence masking is used in the masked multi-head attention layer to avoid exposure to future information. For each token within a SMILES string, the transformer computes the attention weight as follows:

$$Attention(Q, K)V = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right)V,$$

where $Q, K,$ and $V$ are the queries, keys, and values, and have the corresponding dimensions $d_k, d_k,$ and $d_v$. As the multi-head attention projects the queries, keys, and values $h$ times, the transformer jointly attends to features from different strings:

$$\text{MultiHead}(Q, K, V) = [\text{Head}_1, ..., \text{Head}_h]W,$$

where $W, W_Q, W_K,$ and $W_V$ are the weight matrices.

### 3.3 Discriminator Training Process

Let $\{x_1, x_2, ..., x_T\}$ and $\{y_1, y_2, ..., y_T\}$ represent a real molecule and generated sequence, respectively, of the fixed length $T$, where $x_i, y_t \in \mathbb{R}^k$ are $k$-dimensional token embedding vectors. The source matrix $X_{1:T}$ and target matrix $Y_{1:T}$ are then expressed as

$$X_{1:T} = x_1 \oplus ... \oplus x_T, \quad Y_{1:T} = y_1 \oplus ... \oplus y_T,$$

where $\oplus$ is the concatenation operator and $X_{1:T}, Y_{1:T} \in \mathbb{R}^{T \times k}$. By applying a convolutional operation with a window size of $t$ tokens, we then obtain a new feature map:

$$c_{ij} = \text{ReLU}(\omega_j \odot X_{i:t+i+t-1} + b),$$

where $\omega_j$ is the $j$-th kernel and $\omega_j \in \mathbb{R}^{t \times k}$, the $\odot$ operator represents element-wise summation of the production, and $b$ is the bias of the convolutional layer. Note that kernels with various window sizes extract various latent features. Finally, the features are selected by the max-pooling technique:

$$\tilde{c}_j = \max\{c_{j1}, ..., c_{jT-t+1}\}.$$
3.4 Policy Gradient Training Process

The sampling process of molecules represented by SMILES strings is not differentiable. To solve this problem, we adopted an RL approach. Specifically, we can define $G_\theta(Y_{1:T}|X_{1:T})$ as the stochastic policy of the agent and let $s_0$ and $a$ represent the initial state and action, respectively. Under the guidance of the policy gradient, the generator generates sequences by maximizing the expected reward score:

$$J(\theta) = \sum_{Y_{1:T}} G_\theta(Y_{1:T}|X_{1:T}) \cdot R((X_{1:T}, Y_{1:T-1}), y_T), \quad (8)$$

where the action-value function $R(s_0, a)$ denotes the expected reward of starting from state $(X_{1:T}, Y_{1:T-1})$ and taking the action $y_T$. The action-value function is calculated in three main parts: the discriminator probability of the current input sequence, which is determined from the original SMILES dataset; the value of the desired chemical property of the current input sequence $O(Y_{1:T})$; and the penalty function for a generated sequence $Y_{1:T}$ with low diversity. Note that both $O(Y_{1:T})$ and $P(Y_{1:T})$ are valued between 0 and 1 inclusive. $O(Y_{1:T})$ refers to objective scores of the generated SMILES strings, such as the quantitative estimate of drug-likeness (QED) scores. $P(Y_{1:T})$ refers to the penalty for generating non-unique SMILES strings and is calculated by

$$P(Y_{1:T}) = \frac{\text{# of unique SMILES}}{\text{# of SMILES} \times \text{# of repeated } Y_{1:T}}, \quad (9)$$

When the score is zero, the generated sequence represents an invalid molecule or already exists. Otherwise, the sequence describes a valid molecule with the desired chemical property. The reward function is then given by

$$R((X_{1:T}, Y_{1:T-1}), y_T) = \lambda \cdot D_\phi(Y_{1:T}) + (1 - \lambda) \cdot O(Y_{1:T}) \cdot P(Y_{1:T}), \quad (10)$$

where $\lambda$ is a hyperparameter that adjusts the learning ratio of Naïve RL and SeqGAN. However, the above function can only provide a reward score of the entire string. When the generated string $Y_{1:T}$ is incomplete, the values of $D_\phi(Y_{1:T})$, $O(Y_{1:T})$, and $P(Y_{1:T})$ are nonsensical. The action-value function of the subsequence generated at each intermediate time step is calculated by using a Monte Carlo (MC) search under the policy $G_\theta$. The search is continued until the sequence length reaches $T$. To generalize the MC search, we can repeat the MC search $N$ times for each partial sequence:

$$MC^{G_\theta}((X_{1:T}, Y_{1:t}), N) = \{Y_{1:T}^1, Y_{1:T}^2, ..., Y_{1:T}^N\}, \quad (11)$$

where $Y_{1:t}^n$ represents the current partial sequence and $Y_{1:t}^n = Y_{1:t}^n$ is sampled via the policy $G_\theta$. For an $N$-times MC search, we obtain $N$ samples from the discriminator. Finally, the $N$ rewards for the partial sequence are averaged as

$$\mathcal{R}_n((X_{1:T}, Y_{1:t-1}), y_t) = \frac{1}{N} \sum_{n=1}^N R((X_{1:T}, Y_{1:t-1}^n), y_T), \quad (12)$$

$$Y_{1:T}^n \in MC^{G_\theta}(Y_{1:t}; N) \quad \text{if } t < T, \quad (13)$$

Algorithm 1 shows the MC search process with the stochastic rollout policy $G_\theta$.

By using the discriminator as a reward function to guide the update of the generator, we can boost the number of high-quality sequences. We can then retrain the discriminator by using realistic generated sequences and the real molecule dataset. The loss function of the discriminator is given by

$$\min_{\theta} -E_{x \sim P_{data}(x)}[\log D_\phi(x)] - E_{y \sim G_\theta}[\log (1 - D_\phi(y))]. \quad (14)$$

Following the training of the discriminator, we retrain the generator. The gradient of $J(\theta)$ is derived as

$$\nabla_\theta J(\theta) \simeq \frac{1}{T} \sum_{t=1}^T \sum_{y_t} \mathcal{R}_n((X_{1:T}, Y_{1:t-1}), y_t) \quad (15)$$

Algorithm 2 outlines the proposed TransORGAN model. First, $G_\theta$ is pre-trained on real SMILES strings by using maximum-likelihood estimation. Next, $D_\phi$ is pre-trained by using binary cross-entropy. Finally, $G_\theta$ and $D_\phi$ are alternately trained by using the policy gradient method. The training phase also uses the teacher-forcing technique to ensure the stability of training.

4 Experiments

We conducted experiments to compare the performance of our proposed TransORGAN with those of RNN-based molecular generation models. The test data were a subset of the ZINC chemical dataset [Ramakrishnan et al., 2014], which contains 134,000 molecules represented by SMILES strings. Our subset comprised 5,000 randomly selected molecules with up to nine heavy atoms (e.g., carbon (C), oxygen (O), nitrogen (N), and fluorine (F)) within the GDB-17 universe of 166.4 billion molecules [Ruddigkeit et al., 2012].
vocabulary size was 45. Note that the maximum length of molecules in [Guimaraes et al., 2017] was set to 51; in this paper, this value is set to 75. We compared TransORGAN against the following models: Random Sampler [Gao and Coyle, 2020], SMILES LSTM [Guimaraes et al., 2017], Naïve RL, and ORGAN [Guimaraes et al., 2017].

4.1 Desired Chemical Properties
The chemical properties are scored in a range from zero (unfavorable) to one (very favorable). Solubility describes the degree of hydrophilicity of a molecule. The log octanol-water partition coefficient (logP) is defined as the logarithm of the ratio of concentrations of a substance in a mixture of two solvents, octanol and water. Drug-likeness uses QED [Bickerton et al., 2012] to describe the likelihood of a compound being a drug. Synthesizability uses the synthetic accessibility score [Ertl and Schuffenhauer, 2009] to measure the difficulty of molecular synthesis.

4.2 Hyperparameter Settings
The generator of TransORGAN is a transformer architecture. We set the dimension of the word embedding to 16 and the dropout rate to 0.2. The encoder and decoder each had four heads and two stacked layers. The generator was pre-trained over 100 epochs by maximum likelihood estimation (MLE). The dimension of the word embedding was 32 for the discriminator. We set the number of kernels as 1, 3, 5, 7, and 9; the kernel size as 20, 30, 40, 50, and 60; and the dropout rate to 0.75. In the pre-training phase, the discriminator was pre-trained over ten epochs. In addition, we set the tradeoff between maintaining the likelihood and RL as \( \lambda = 0.5 \). The MC search time \( N \) was set to 16. All experiments were performed by using Pytorch version 1.8.1.

4.3 Evaluation Measures
Validity was defined as the ratio of chemically valid molecules among all generated molecules. Uniqueness was defined as the number ratio of unique molecules among chemically valid samples. Novelty was defined as the ratio of valid molecules that were absent in the training dataset and the set of chemically valid molecules. Diversity was used to quantify the chemical diversity of the generated molecules. The similarity of chemical structures between molecules was calculated according to the Tanimoto coefficient [Tanimoto, 1968] based on MorganFingerprint [Cereto-Massagué et al., 2015]. All of these statistics had a range from zero to one. A larger value indicated a better performance.

4.4 Performance Evaluation
Objective-reinforced methods aim to create compound structures with desired chemical properties, which are critical in drug discovery. Table 1 demonstrates the comparative results of the objective-reinforced methods. In the case of solubility, Naïve RL scored the highest validity (84.82%), but the lowest uniqueness (46.55%) and diversity (0.339) among the three models. Although TransORGAN had lower validity (74.03%) than Naïve RL, it scored much higher than the other two models in terms of uniqueness, novelty, and diversity. This result is reasonable because Naïve RL seeks only valid molecules without considering other factors. Similar to Naïve RL, ORGAN tries to optimize the tradeoff between validity, uniqueness, and diversity, so it generates valid molecules at the expense of uniqueness and diversity. Repetitive molecules are not desired in practice. Moreover, Naïve RL and ORGAN generated molecules containing many carbon atoms but few other atoms. This is why Naïve RL and ORGAN achieved higher validity than TransORGAN but lower scores for the other three measures. In the case of drug-likeness and synthesizability, ORGAN scored the highest validity (81.59% and 86.60%, respectively) among the three models but the lowest uniqueness (46.58% and 33.36%, respectively).

In drug discovery, chemists hope to generate molecules that are not only valid but also novel. If the generated molecules already exist in the original dataset, the generation is meaningless. TransORGAN achieved the highest uniqueness, novelty, and diversity while maintaining relatively high validity. TransORGAN generated 3,327, 3,326, and 3,367 novel molecules with the desired solubility, drug-likeness, and synthesizability. Compared with Naïve RL, TransORGAN improved the solubility, drug-likeness, and synthesizability scores by 69.66%, 10.90%, and 115.70%, respectively. Compared with ORGAN, it improved the scores by 40.91%, 75.33%, and 133.50%, respectively.

To evaluate the drug-likeness of generated molecules, we measured the QED scores. Figure 2 shows the QED distributions when the objective was drug-likeness. Naïve RL and

---

**Algorithm 2: Pre/training for TransORGAN**

```c
// Pre-train the Generator
for i = 1 to Pre_Gen do
    Update Gθ based on maximum likelihood.

// Pre-train the Discriminator
for i = 1 to Pre_Dis do
    Update Dθ based on binary cross entropy.

// Adversarial training
for i = 1 to Adversarial epochs do
    for j = 1 to Generator epochs do
        // Generate samples
        Y_1:T = Gθ, sample(X_1:T);
        Calculate the expected reward: \( \overline{R}(Y_1:T) \);
        Update the gradient of Gθ using \( \overline{R}(Y_1:T) \);
        // Produce variant SMILES.
        (X, \tilde{X}) = Variant(X);
        // Teacher-forcing training
        Update the gradient of Gθ using
        \( \nabla_{X_1:T, \tilde{X}_1:T} (X_1:T, \tilde{X}_1:T) \);
    for k = 1 to Discriminator epochs do
        Update Dθ based on (\tilde{X}_1:T, Y_1:T)
```
**Table 1: Evaluation results of generated molecules with objective reinforcement.**

<table>
<thead>
<tr>
<th>Objective</th>
<th>Method</th>
<th>Validity</th>
<th>Uniqueness</th>
<th>Novelty</th>
<th>Diversity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solubility</td>
<td>Naïve RL</td>
<td>4,234 (84.82%)</td>
<td>1,971 (46.55%)</td>
<td>1,961 (99.49%)</td>
<td>0.339</td>
</tr>
<tr>
<td></td>
<td>ORGAN</td>
<td>3,343 (66.97%)</td>
<td>2,381 (71.22%)</td>
<td>2,361 (99.16%)</td>
<td>0.358</td>
</tr>
<tr>
<td></td>
<td>TransORGAN</td>
<td>3,654 (74.03%)</td>
<td><strong>3,327 (91.05%)</strong></td>
<td><strong>3,327 (100.0%)</strong></td>
<td><strong>0.650</strong></td>
</tr>
<tr>
<td>Drug-likeness</td>
<td>Naïve RL</td>
<td>3,986 (79.85%)</td>
<td>1,897 (46.58%)</td>
<td>1,897 (100.0%)</td>
<td>0.338</td>
</tr>
<tr>
<td></td>
<td>ORGAN</td>
<td>4,073 (81.59%)</td>
<td>1,897 (46.58%)</td>
<td>1,897 (100.0%)</td>
<td>0.338</td>
</tr>
<tr>
<td></td>
<td>TransORGAN</td>
<td>3,589 (72.71%)</td>
<td><strong>3,327 (92.67%)</strong></td>
<td><strong>3,327 (100.0%)</strong></td>
<td><strong>0.624</strong></td>
</tr>
<tr>
<td>Synthesizability</td>
<td>Naïve RL</td>
<td>3,874 (77.60%)</td>
<td>1,561 (40.29%)</td>
<td>1,561 (100.0%)</td>
<td>0.365</td>
</tr>
<tr>
<td></td>
<td>ORGAN</td>
<td>4,323 (86.60%)</td>
<td>1,442 (33.36%)</td>
<td>1,442 (100.0%)</td>
<td>0.391</td>
</tr>
<tr>
<td></td>
<td>TransORGAN</td>
<td>3,668 (74.31%)</td>
<td><strong>3,367 (91.79%)</strong></td>
<td><strong>3,367 (100.0%)</strong></td>
<td><strong>0.656</strong></td>
</tr>
</tbody>
</table>

Bold values indicate the maximum values in the objective-reinforced methods.

**Figure 2: Distributions of the QED scores for generated molecules.**

**Table 2: Comparison of TransORGAN with baseline methods.**

<table>
<thead>
<tr>
<th>Method</th>
<th>Validity</th>
<th>Uniqueness</th>
<th>Novelty</th>
<th>Diversity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Sampler</td>
<td>100.0%</td>
<td>61.54%</td>
<td>0.0%</td>
<td>0.638</td>
</tr>
<tr>
<td>SMILES LSTM</td>
<td>42.81%</td>
<td>98.41%</td>
<td>98.48%</td>
<td>0.423</td>
</tr>
<tr>
<td>JT-VAE</td>
<td>100.0%</td>
<td>13.94%</td>
<td>99.43%</td>
<td>0.607</td>
</tr>
<tr>
<td>Graph-AF1*</td>
<td>72.26%</td>
<td>84.80%</td>
<td>100.0%</td>
<td>0.786</td>
</tr>
<tr>
<td>Graph-AF10*</td>
<td>67.27%</td>
<td>99.44%</td>
<td>100.0%</td>
<td>0.691</td>
</tr>
<tr>
<td>Character-VAE</td>
<td>73.34%</td>
<td>99.18%</td>
<td>100.0%</td>
<td>0.392</td>
</tr>
<tr>
<td>Grammar-VAE</td>
<td>76.36%</td>
<td>99.55%</td>
<td>100.0%</td>
<td>0.451</td>
</tr>
<tr>
<td>TransORGAN</td>
<td><strong>75.52%</strong></td>
<td><strong>94.64%</strong></td>
<td><strong>100.0%</strong></td>
<td><strong>0.682</strong></td>
</tr>
</tbody>
</table>

The superscripts 1 and 10 for Graph-AF indicate the minimum length of the SMILES strings.

**5 Conclusions**

We proposed TransORGAN with the aim of generating high-quality molecules represented as SMILES strings in an objective-reinforced manner. By leveraging the joint training of a sequence-based GAN model and an RL objective, the proposed model generates molecules with high degrees of uniqueness, novelty, and diversity.

One limitation of TransORGAN is the difficulty of optimizing the values for the hyperparameters $\lambda$ and $N$. In future work, we will investigate the effects of changing $\lambda$ and $N$. Because $\lambda$ controls the tradeoff between MLE and RL, it may also affect the performance of the molecular generation. When $\lambda$ is small, TransORGAN strongly believes in the ability of the discriminator. With increasing $\lambda$, TransORGAN becomes more focused on generating valid molecules. The number of sample times $N$ in the MC search may also influence the performance. Intuitively, a small $N$ may lead to wrongly generated intermediate tokens, whereas a large $N$ increases the computational time. We believe that the proposed models will help chemists produce meaningful new drugs.
Acknowledgements
This research was supported by AMED under Grant Number JP21nk0101111 and JSPS KAKENHI [grant number 20H05797].

References


