Sequence Studies for Text Data Augmentation and Molecular Generation

Dissertation submitted in partial fulfillment for the
degree of Ph.D. in Informatics and Data Science

Huidong Tang

Under the supervision of
Professor Yasuhiko Morimoto

Social Computing Laboratory,
Graduate School of Advanced Science and Engineering,
Division of Advanced Science and Engineering Informatics and Data Science
Program,
Hiroshima University, Higashi-Hiroshima, Japan

April 2024

Acknowledgement

I appreciate my supervisors, collaborators, colleagues, and family.

Abstract

This thesis studies sequence modeling for text data augmentation and molecular generation. Sequence modeling is a widely studied research area, including stock price and weather data prediction, gene sequencing, Natural Language Processing (NLP), and molecular generation. These research topics cover economics, environment, biology, linguistics, and chemistry, which are tightly linked to the development of society.

Among these foundations, natural language is the cornerstone of civilization, facilitating communication and storing knowledge. To solve language-related problems with the help of computational techniques, NLP, which consists of various tasks such as text classification, entity extraction, text summarization, and text generation, is being developed. Text classification is one of the fundamental NLP tasks, that aims to categorize a text into one or more classes. However, the robustness of such a task remains a concern, as the predictions can be manipulated by adding perturbations. To address this concern, we propose three data augmentation methods based on word substitution, combining synonyms, antonyms, and sentimentally related words for the robustness enhancement of the text classification task. We attempted to generate samples that differed in semantics from the training data to improve the robustness. We evaluated our methods on four publicly available datasets using text adversarial attack techniques, and the experimental results validated the robustness enhancement.

On the other hand, we also studied molecular generation for drug discovery using Generative Adversarial Networks (GANs). Modeling molecular generation using the Simplified Molecular Input Line Entry System (SMILES) as a token-level sequence generation is straightforward. However, naively adopting the cumulative reward for the token-level sequence generation is time-consuming and incompatible with the SMILES nature. To address these limitations, we introduced an efficient reward function that combines moment and global rewards, along with the information entropy maximization, as an alternative to the cumulative reward. The combination of moment and global rewards reduces the training time and ensures molecular consistency, and the information entropy maximization allows for diverse explorations and avoids the mode collapse problem common in GANs. Our Enhanced Actor-critic Reinforcement Learning (RL) agent-driven GAN, EarlGAN, can generate molecules with a highly balanced performance. Our extensive evaluation experiments validated the effectiveness of our model.

Contents

1 Introduction

We studied two topics related to sequence modeling: data augmentation for enhancing the robustness of the text classification task and molecular generation with an efficient reward function.

NLP is being developed to solve language-related problems, among which text classification is one of the fundamental tasks, including spam detection [1], topic classification [2], and sentiment classification [3]. It helps reduce labor costs by using computers to categorize text automatically, and Language models (LMs) are particularly helpful in solving such a task. However, the robustness of the LMs for the text classification task remains a concern [4, 5, 6]. These studies demonstrated that LMs are vulnerable to adversarial attacks, casting shadows on the applications. The adversarial attacks add perturbations, such as character or word substitutions, to the predicted text, altering the prediction and compromising the reliability of the prediction results. To enhance the robustness of LMs for the text classification task, previous studies [4, 6, 7] focused on adversarial training. Adversarial training adds adversarial samples generated by adversarial attacks to the training data, providing features of the adversarial samples. However, adversarial samples require gradient computation or additional computations such as importance score [4], until the prediction of the sample changes. Such computations are based on additional backpropagation or multi-trial, which are expensive.

In such a context, we paid attention to data augmentation techniques [8, 9, 10, 11, 12, 13, 14, 15] to provide samples with similar functions of adversarial samples at lower cost. We assumed that large changes in semantics can also improve the robustness. To induce relatively large semantic changes, we used sentimentally related words for data augmentation. Specifically, we proposed three data augmentation methods for robustness enhancement [16]. Cognate-based methods combine synonyms and sentimentally similar words; Antonym-based methods use antonyms; and Antipode-based methods combine antonyms and sentimentally opposite words.

To evaluate our methods, we measured the performance and robustness results; the former is based on accuracy and F1 score, and the latter is based on the accuracy under attack and the attack success rate. The accuracy under attack is the ratio of the number of samples that resist attacks (maintain correct predictions) to the total number of samples. The attack success rate is the ratio of the number of samples predicted correctly to incorrectly to the number of samples originally predicted correctly. We evaluated our methods on four datasets: AG News [17], TREC [18, 19], SUBJ [20], and SMS Spam [21]. We selected DistilBERT [22] as the target victim model, which is attacked by the adversarial attacks [4, 7]. Our evaluation experiments validated the effectiveness of our methods and improved the performance and robustness of the target victim model.

Drug discovery ensures the sustainability and development of modern healthcare and human well-being. It is aimed at discovering new chemical compounds with healing effects. However, to develop a new drug costs 10-15 years and 2 billion U.S. dollars on average [23]. Artificial Intelligence (AI) has attracted the attention of the pharmaceutical industry [24] for time and cost reduction. Among AI applications, deep generative models have made remarkable progress, such as ChatGPT [25] and DALLE-3 [26] in NLP and Computer Vision (CV) areas.

Recently, deep generative models for drug discovery have been increasingly studied, including Variational AutoEncoders (VAEs) [27, 28, 29], GANs [30, 31], and diffusion models [32]. In this study, GANs were chosen as the basic architecture for rapid training. On the one hand, although GANs have the mode collapse problem, the samples they generate are more realistic than those generated by VAEs. This is because the latent space of VAEs is generally too small, resulting in a lack of express capability. On the other hand, despite the powerful performance diffusion models, the multi-step denoising process takes more time to train than GANs.

SMILES [33] is a string form of molecular structure with simpler information, which reduces the computational cost. However, direct modeling of SMILES-based molecular generation using GANs is impractical due to the discrete nature of SMILES. A GAN consists of two components: a generator and a discriminator. The generator samples from the training data distribution and tries to generate samples to fool the discriminator, while the discriminator tries to distinguish the generated samples from the real samples and updates the generator using true/false discrimination. Such a competitive cycle guides the learning of GANs. The subtle changes in the discrimination probability cannot be reflected as a binary classification, leading to the gradient vanishing problem. Therefore, we directly used the discrimination probability as the reward for the generator update instead of using the discrimination true/false result.

Furthermore, we followed [34, 35] to model SMILES generation at the token-level for dense rewards, which provide more efficient learning signals early in training. However, the cumulative reward for generating SMILES at the token-level is time-consuming and incompatible with the inherent nature of SMILES. The cumulative reward assumes that the influence between tokens is inversely proportional to the distance [34], which is not absolutely correct in SMILES.

To address these two problems, we introduced an efficient reward function instead of the cumulative reward for token-level SMILES generation [36, 37]. We combined the moment reward for time efficiency and the global reward [38] to ensure molecular consistency and information entropy maximization for diverse exploration. Our reward function specializes in SMILES generation and is an alternative to the cumulative reward, which is time-consuming and incompatible with the nature of SMILES. We performed extensive evaluation experiments validating the effectiveness of EarlGAN for SMILES generation, and the chemical properties of the generated SMILES align with the training data distribution.

The organization of this thesis is as follows: Chapter 2 presents the preliminary for this thesis; Chapters 3 and 4 introduce text data augmentation techniques for enhancing the robustness of the text classification task and an efficient reward function for molecular generation and Chapter 5 concludes this thesis.

2 Preliminary

2.1 Natural Language Processing (NLP) and Text Classification

NLP deals with language-related tasks, such as text classification, entity extraction, text generation, and so on. Among these tasks, text classification is to categorize a text into one or more classes.

2.2 Language Models (LMs)

Researchers have recently been studying the application of neural networks to address such NLP tasks. Among these applications, LMs have made progress, such as BERT [39] and DistilBERT [22]. Most LMs are based on Transformer architecture [40] and are typically trained using word prediction tasks.

2.3 Adversarial Training

Adversarial training aims to improve the robustness of machine learning models so that they perform well not only on data similar to the training data but also on unknown samples. Typically, for the text classification task, this involves perturbing the training data to induce the model to make an incorrect prediction. This can be done at the character-level, word-level, sentence-level, or a combination of these operations. The samples that effectively induced the incorrect prediction are then input as new training samples.

2.4 Data Augmentation

Data augmentation is typically used to improve the performance of machine learning models when data is scarce, such as when data is difficult to obtain or labeling is costly. In such cases, data augmentation techniques are used to learn more samples and reduce unknown data.

2.5 Generative Adversarial Networks (GANs)

GANs are one of the dominant deep generative models, a GAN consists of two components: the generator and the discriminator. They play a max-min game: the generator tries to generate samples aligned with the training data distribution and fool the discriminator into predicting the generated samples as real samples; the discriminator tries to distinguish the realness of the input samples. The value function of GAN [41] is described as follows:

$$\begin{aligned}
&\min_{\theta}\max_{\phi} V(G_\theta, D_\phi) = \\
&\mathbb{E}_{x\sim p_{data}(x)}[logD_\phi(x)]+ \\
&\mathbb{E}_{z\sim p_z(z)}[log(1-D_\phi(G_\theta(z)))],
\end{aligned} \tag{2.1}$$

where G_θ and D_ϕ are the generator and discriminator, θ and ϕ are their parameters, respectively. $p_{data}(x)$ and $p_z(z)$ are the real data and input noise distribution, respectively.

2.6 Monte Carlo Tree Search (MCTS)-based Reinforcement Learning (RL)

RL is used to guide a computational agent to perform a task. RL does not require a static dataset for the training, the agent can communicate with the environment to obtain the experiment as a dataset. The key factors are the agent, environment, actions, observations, and rewards. The agent performs various actions with the environment and observes the rewards returned from the environment, this process is performed iteratively until the training is completed. Recently, reinforcement learning has been increasingly combined with deep learning, called deep reinforcement learning, to perform complicated tasks, such as self-driving, robotics, and game-playing.

Among these applications, GANs with MCTS-based RL algorithms are commonly used for sequence generation, including molecular generation. MCTS-based RL algorithms mimic the tree traversal architecture to obtain the reward of each node, which is calculated based on the results at the end of the sequence. To stabilize the training process, MCTS-based RL algorithms require a large sampling number for each node's reward calculation due to the inefficient sampling manner.

2.7 Long-Short Term Memory (LSTM)

LSTM [42] is a neural network architecture proposed to solve the long-distance dependency problem in sequence modeling. When a long sentence is fed into a neural network, the network cannot process word relationships that are too far apart. LSTM [42] mainly uses the forget gate and the input gate to maintain the long-distance dependencies. The forget gate determines which information should be removed, and the input gate determines which information should be updated. Finally, these two components are combined to obtain the cell state and output autoregressively. Generally, LSTM [42] can be mathematically described as follows:

$$h_t = o_t \times tanh(C_t), \tag{2.2}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \tag{2.3}$$

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t, \tag{2.4}$$

$$\tilde{C}_t = tanh(W_C[h_{t-1}, x_t] + b_C), \tag{2.5}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \tag{2.6}$$

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f), \tag{2.7}$$

where h_t is the output result, x_t is the input, h_{t-1} is the previous output result, σ is the sigmoid layer, b_o, b_C, b_i and b_f are the learnable bias terms, f_t and i_t are the forget and input gate, respectively. W_o, W_C, W_i and W_f are the learnable weights. C_t is the current cell state, C_{t-1} is the previous cell state, $\tilde{C}_t$ is the candidate current cell state.

3 Data Augmentation for Enhancing the Robustness of the Text Classification Task

3.1 Related Work

Various adversarial training methods have been proposed to improve robustness. TEXTFOOLER [4] replaces the important words based on the importance score in the text with semantically similar and grammatically correct ones for such training. CLARE [6] employs a pre-trained LM to perform replace, insert, and merge actions to achieve the adversarial training. PWWS [7] is a synonym-based word replacement method that considers both the word saliency and the classification probability. Despite successful robustness improvements, these methods require additional computations such as gradient or importance scores.

On the other hand, data augmentation methods are designed to improve accuracy, especially for small datasets. EDA [8] performs one of the four-word manipulations: random synonym replacement, random word deletion, random word position swap, and random synonym insertion. AEDA [9] is a successor to EDA [8] that randomly inserts punctuation marks into the text to improve performance. Data Boost [10] and LAMBADA [11] both use pre-trained LMs for this end; the former combines the LM with RL for guidance, while the latter adds a data filtering process for high-quality data generation. UDA [12] transfers data augmentation methods from supervised to semi-supervised settings, expanding the applications of data augmentation. Contextual Augmentation [13] replaces random words in sentences based on a bidirectional language model prediction. [14] employs dependency trees to perform "crop" and "rotate". [15] optimizes multiple data augmentation policies to find an effective "path" for text data augmentation. Although these data augmentation methods have proven to be effective for performance gains, especially for small datasets, none of them pay attention to robustness.

3.2 Data Augmentation Methods

We proposed three types of data augmentation methods to improve the robustness of LMs: Cognate-based, Antonym-based, and Antipode-based stochastic word replacement methods.

- Cognate-based Methods: We stochastically replaced words with synonyms and sentimentally similar words. Sentimentally similar words were defined as words that have the same sensitivity, attitude, temper, and introspection scores as the target word. These scores represent twenty-four basic emotions: ecstasy, joy, contentment, melancholy, sadness, grief, bliss, calmness, serenity, annoyance, anger, rage, delight, pleasantness, acceptance, dislike, disgust, loathing, enthusiasm, eagerness, responsiveness, anxiety, fear, and terror. For example, a high introspection score represents joy, while a low introspection score represents sadness.
- Antonym-based Methods: We stochastically replaced words with antonyms.
- Antipode-based Methods: We stochastically replaced words with antonyms and sentimentally opposite words. Sentimentally opposite words were defined as words that have the opposite sensitivity, attitude, temper, and introspection scores as the target word.

We applied three grammatical constraints to our methods to validate the effect of grammar rules for data augmentation, resulting in nine variants.

- Cognate1 (resp. Antonym1, Antipode1): We did not place any restrictions on these methods.
- Cognate2 (resp. Antonym2, Antipode2): We applied the constraint that the candidate replacement words must have the same Part-Of-Speech (POS) tag as the target word, with the exception that nouns and verbs can be replaced with each other.
- Cognate3 (resp. Antonym3, Antipode3): We applied the constraint that the candidate replacement words must have the same POS tag as the target word.

We implement these methods using WordNet [43] and SenticNet [44]. We listed some augmented data samples from AG News [17] as follows:

- Original: "Bangladesh paralysed by strikes Opposition activists have brought many towns and cities in Bangladesh to a halt, the day after 18 people died in explosions at a political rally."
- Cognate-based Methods: "Bangladesh paralysed by strikes Opposition activists have brought many **townsfolk** and cities in Bangladesh to a halt, the day after 18 people **edema** in explosions at a political rally."

Table 3.1: Datasets Description.

Dataset	Task	Number of Classes	Training Size	Evaluation Size	Test Size
AG News [17]	News Topic Classification	4	5k	0.5k	0.5k
TREC [18, 19]	Question Classification	6	4.5k	0.5k	0.5k
SUBJ [20]	Movie Review Subjectivity Classification	2	5k	0.5k	0.5k
SMS Spam [21]	Spam SMS Detection	2	4.5k	0.5k	0.5k

- Antonym-based Methods: “Bangladesh paralysed by strikes Opposition activists have brought **few** towns and cities in Bangladesh to a halt, the day after 18 people died in explosions at a **nonpolitical** rally.”
- Antipode-based Methods: “Bangladesh paralysed by strikes Opposition activists have brought many towns and cities in Bangladesh to a halt, the **night** after 18 people died in explosions at a political **demobilize**.”

3.3 Evaluation Experiments

3.3.1 Datasets

We evaluated our data augmentation methods on AG News [17], TREC [18, 19], SUBJ [20], and SMS Spam [21]. We sampled these datasets, and detailed descriptions are given in Table 3.1.

3.3.2 Evaluation Metrics

We conducted evaluation experiments based on four metrics: accuracy, F1 score, accuracy under attack, and attack success rate. The first two metrics are used for performance evaluation, while the last two metrics are used for robustness evaluation. The larger the values of these evaluation metrics, the more favorable, except for the attack success rate.

The accuracy and F1 score are calculated based on the evaluation confusion matrix given in Table 3.2. True Positive (TP) is the number of samples that are correctly predicted as "positive". False Positive (FP) is the number of samples that are incorrectly predicted as "positive". True Negative (TN) is the number of samples that are correctly predicted as "negative". False Negative (FN) is the number of samples that are incorrectly predicted as "negative". "Positive" and "negative" mean that a sample belongs to or does not belong to a particular class, respectively. For binary classification problems, the accuracy and the F1

Table 3.2: Evaluation Confusion Matrix.

	Positive (Predicted)	Negative (Predicted)
Positive (Actual)	TP	FN
Negative (Actual)	FP	TN

score are calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{3.1}$$

$$F1 = \frac{2Precision \times Recall}{Precision + Recall}, \tag{3.2}$$

$$Precision = \frac{TP}{TP + FP}, \tag{3.3}$$

$$Recall = \frac{TP}{TP + FN}, \tag{3.4}$$

where precision is the percentage of "positive" predictions that are correct, and recall is the percentage of the actual "positive" samples that are predicted as "positive". Although precision and recall are both useful evaluation metrics, they have a trade-off. The F1 score is used for a balanced evaluation.

For multi-class classification problems, we used a macro F1 score that gives equal weight to each class. The accuracy and the macro F1 score $F1_{Macro}$ are defined as follows:

$$Accuracy = \frac{N_C}{N_A}, \tag{3.5}$$

$$F1_{Macro} = \frac{\sum_{n=1}^{N} F1_n}{N}, \tag{3.6}$$

$$F1_n = \frac{2Precision_n \times Recall_n}{Precision_n + Recall_n}, \tag{3.7}$$

$$Precision_n = \frac{TP_n}{TP_n + FP_n}, \tag{3.8}$$

$$Recall_n = \frac{TP_n}{TP_n + FN_n}, \tag{3.9}$$

where N_C is the number of correct predictions, N_A is the number of all predictions, $n$is the n-th class, and N is the total number of classes.

The attack success rate R_{AS} is defined as follows:

$$R_{AS} = \frac{N_I}{N_C}, \tag{3.10}$$

where N_I is the number of samples predicted correctly to incorrectly under attack, and N_C is the number of samples that are originally predicted correctly without attack.

3.3.3 Evaluation Implementation

To validate our methods, we used our data augmentation methods to generate additional training datasets that were merged with the original training datasets. We fine-tuned DistilBERT [22] with these merged training datasets and evaluated the performance.

Then, we used TEXTFOOLER [4] and PWWS (excluding the named entity adversarial swap) [7] to attack these trained models for robustness evaluation. We also compared with EDA [8] and CheckList [5]. All evaluations were repeated three times, and the results were averaged out.

3.4 Results

We presented the evaluation results as the difference between the fine-tuned model with and without data augmentation methods.

3.4.1 Accuracy and F1 Score Results

Figures 3.1-3.4 show the comparison results in terms of accuracy and F1 score on the AG News [17], TREC [18, 19], SUBJ [20], and SMS Spam [21] datasets.

The red bars represent the accuracy, and the blue bars represent the F1 score. For the AG News [17] dataset, all data augmentation methods improved the accuracy and F1 score. **Antipode3** is the most impressive, with increases of 1.87% and 1.91% in accuracy and F1 score, respectively. However, **Antipode1** cannot effectively improve these metrics. This difference is due to the POS constraint, which indicates the importance of data filtering based on grammar rules. For the TREC [18, 19] dataset, all data augmentation methods decreased the performance. The TREC [18, 19] is a relatively simple question classification dataset, and these data augmentation methods may introduce more noise, which decreases the performance. For the SUBJ [20] dataset, **EDA [8]** and **Antonym1** are effective in improving performance. For the SMS Spam [21] dataset, all data augmentation methods can improve the performance, which is similar to the results for the AG News [17] dataset. **CheckList [5]** and **Antipode2** are comparable in performance improvement, the former increased accuracy and F1 score by 0.53% and 1.22%, respectively, while the latter increased them by 0.47% and 1.05%. Based on the observation of the performance results on the AG News [17] and SMS Spam [21] datasets, the Antipode-based methods with appropriate restrictions are relatively effective in improving the performance.

3.4.2 Robustness Results

Figures 3.5-3.12 show the results of the robustness improvement. The red bars represent attacks by TEXTFOOLER [4], and the blue bars represent attacks by PWWS [7]. For the AG News [17] dataset, **Antonym2** and **Antipode2** demonstrated high robustness, especially **Antipode2** with accuracy improvement under attack (2.8% and 8.6%) and attack success rate decrease (-3.0% and

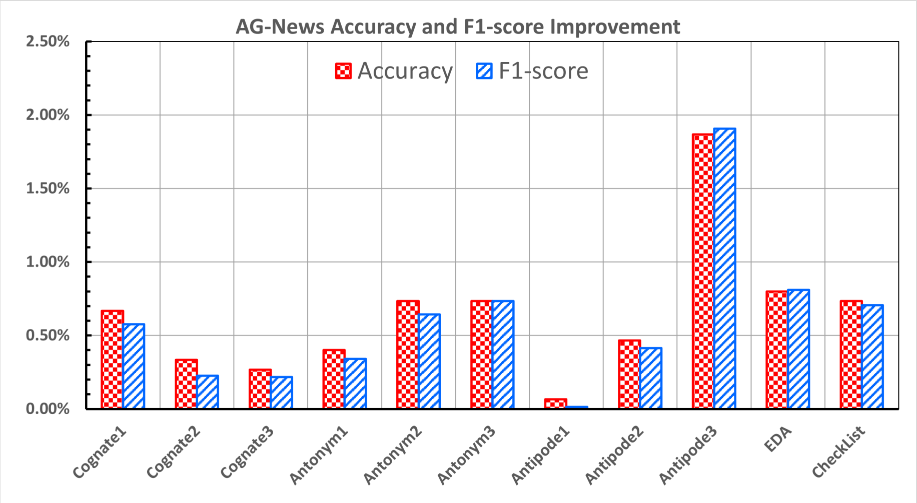


Figure 3.1: AG News Accuracy and F1 Score Improvement.

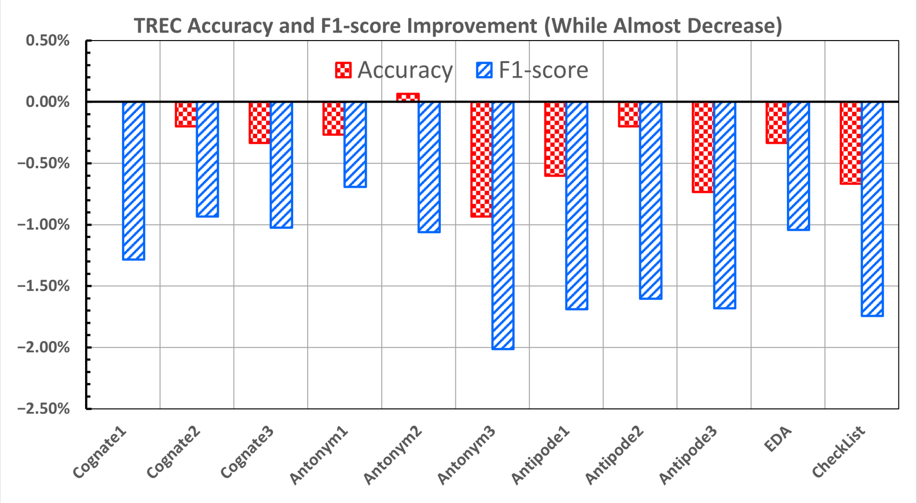


Figure 3.2: TREC Accuracy and F1 Score Improvement.

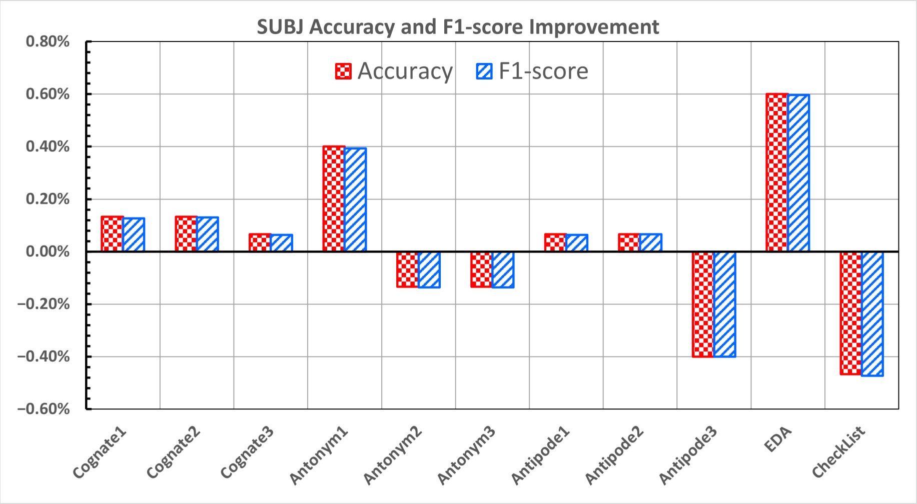


Figure 3.3: SUBJ Accuracy and F1 Score Improvement.

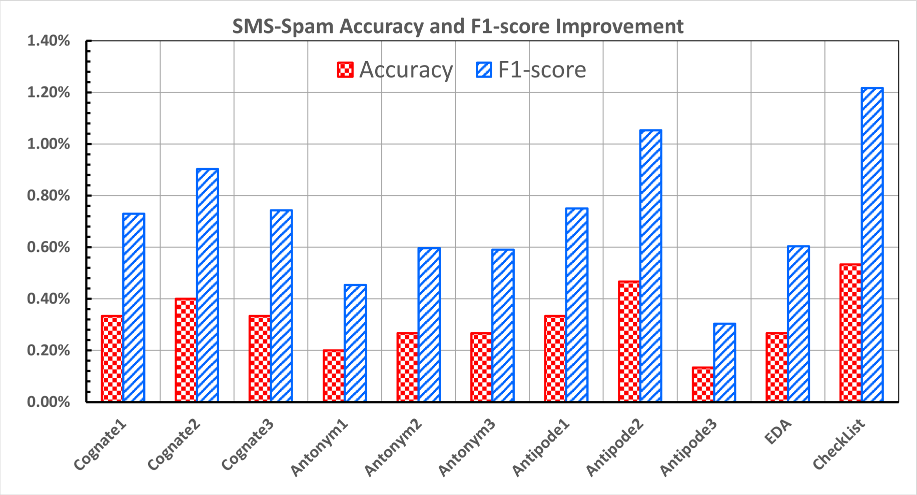


Figure 3.4: SMS Spam Accuracy and F1 Score Improvement.

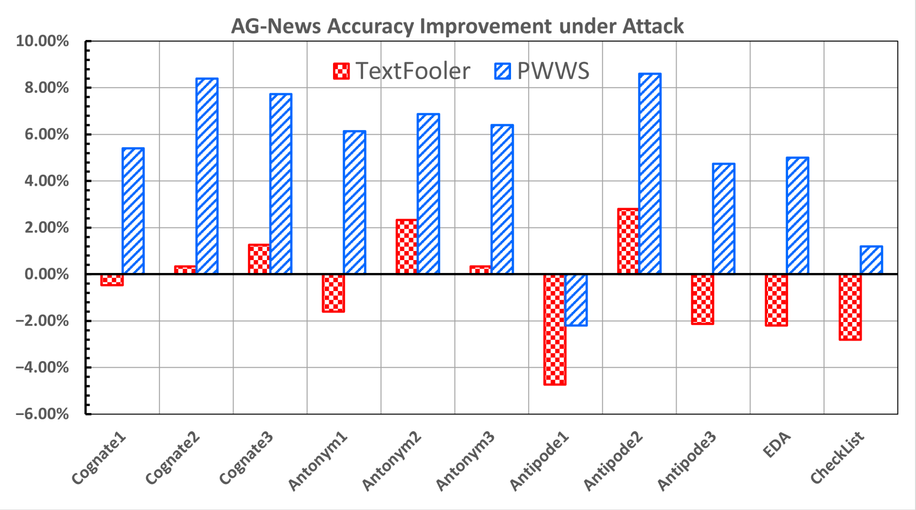


Figure 3.5: AG News Accuracy Improvement under Attack.

-9.34%) by TEXTFOOLER [4] and PWWS [7]. For the TREC [18, 19] dataset, **Antonym1** and **Antonym2** improved the robustness in terms of accuracy under attack (1.07% and 3.8%; 1.67% and 1.27%), and attack success rate (-1.19% and -4.08%; -1.7% and -1.27%). For the SUBJ [20] dataset, **Antonym2** and **Antipode1** worked to increase accuracy under attack (1.13% and 2.33%; 1.0% and 2.53%) and decrease attack success rate (-1.2% and -2.48%; -1.02% and -2.61%). For the SMS Spam [21] dataset, all methods are effective in improving robustness, especially **Antipode2** significantly increased accuracy under attack (16.93% and 11.47%) and decreased attack success rate (-16.68% and -11.15%).

Although the robustness improvement varied depending on the dataset, Antonym-based and Antipode-based methods validated their effectiveness. We hypothesized that learning with antonyms and sentimentally opposite words would allow the model to learn more out-of-distribution samples, thereby improving the robustness.

4 Molecular Generation with An Efficient Reward Function

4.1 Related Work

Deep generative models have emerged in real-world applications in NLP and Computer Vision (CV) areas. Researchers have made efforts to apply these models to drug discovery. GrammarVAE [29] attempts to generate syntactically correct SMILES strings using parse trees, but without regard to semantic meaning. SyntaxVAE [27] improves on GrammarVAE [29] by taking both grammar

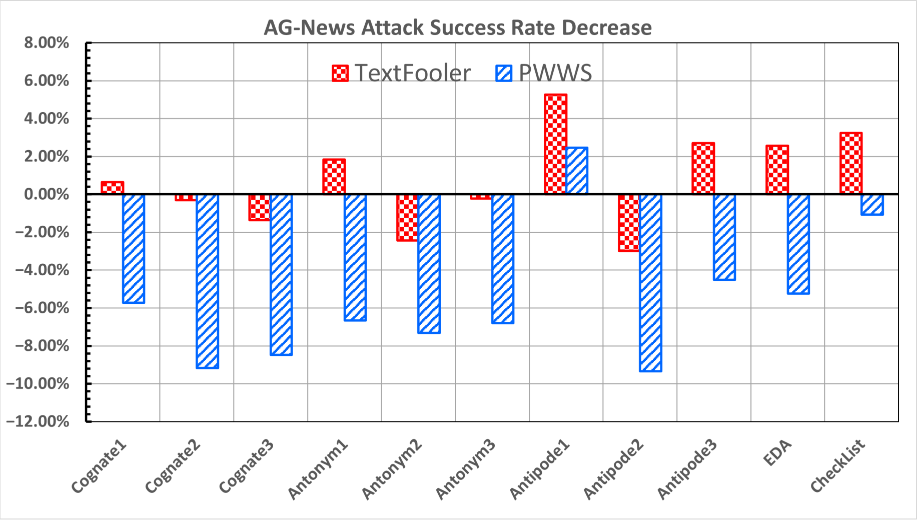


Figure 3.6: AG News Attack Success Rate Decrease.

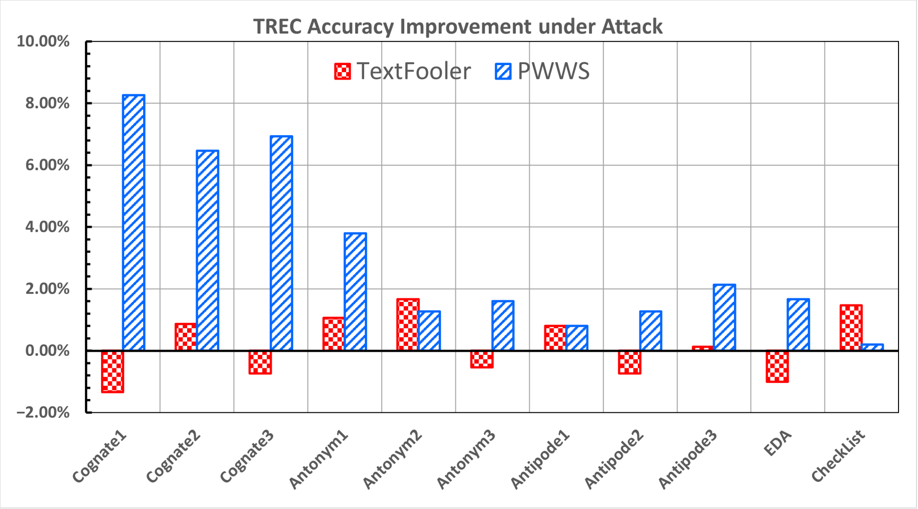


Figure 3.7: TREC Accuracy Improvement under Attack.

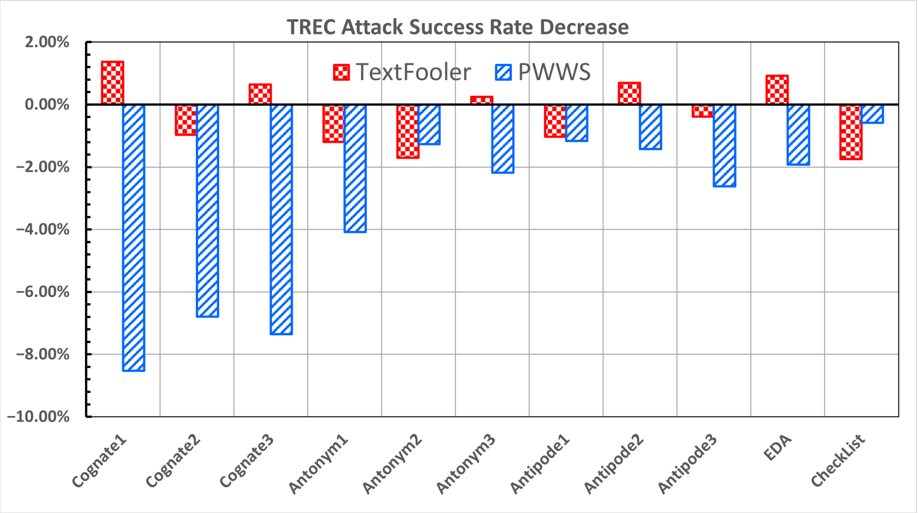


Figure 3.8: TREC Attack Success Rate Decrease.

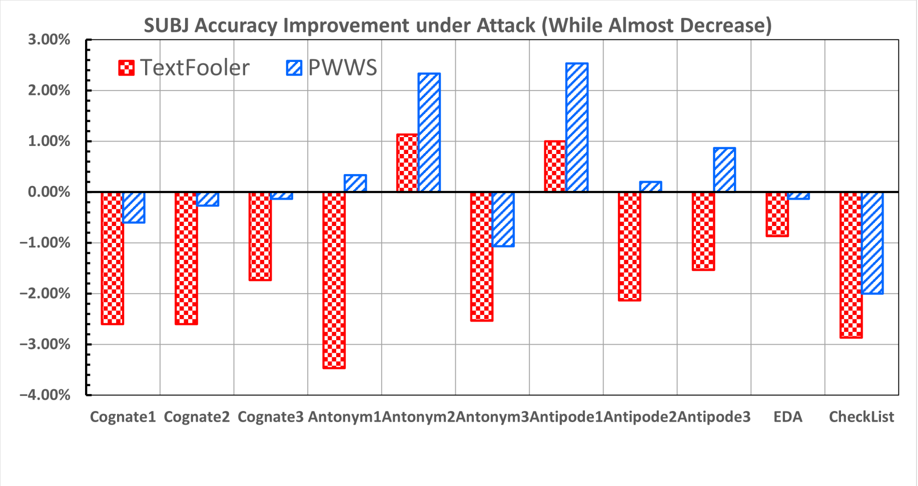


Figure 3.9: SUBJ Accuracy Improvement under Attack.

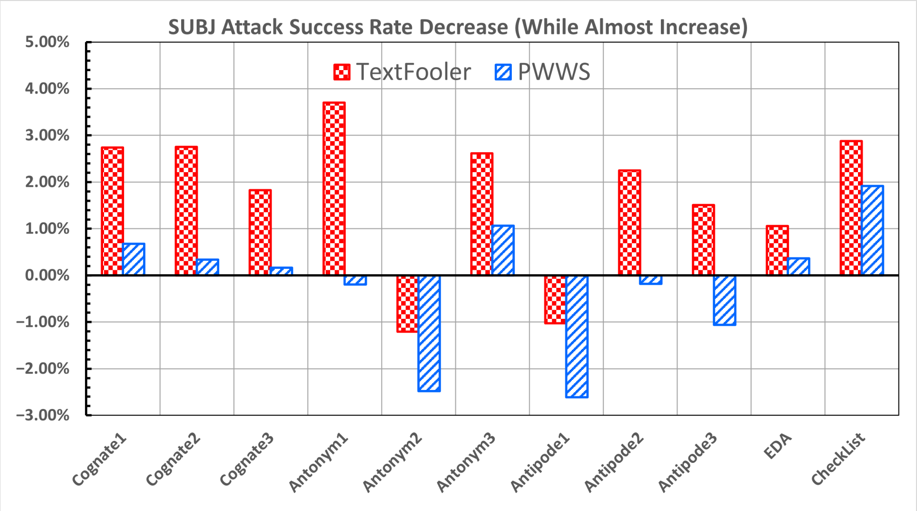


Figure 3.10: SUBJ Attack Success Rate Decrease.

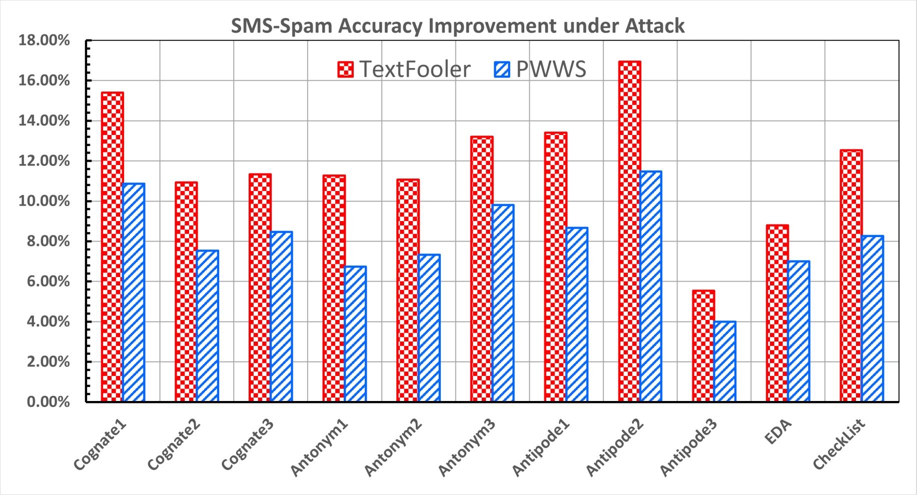


Figure 3.11: SMS Spam Accuracy Improvement under Attack.

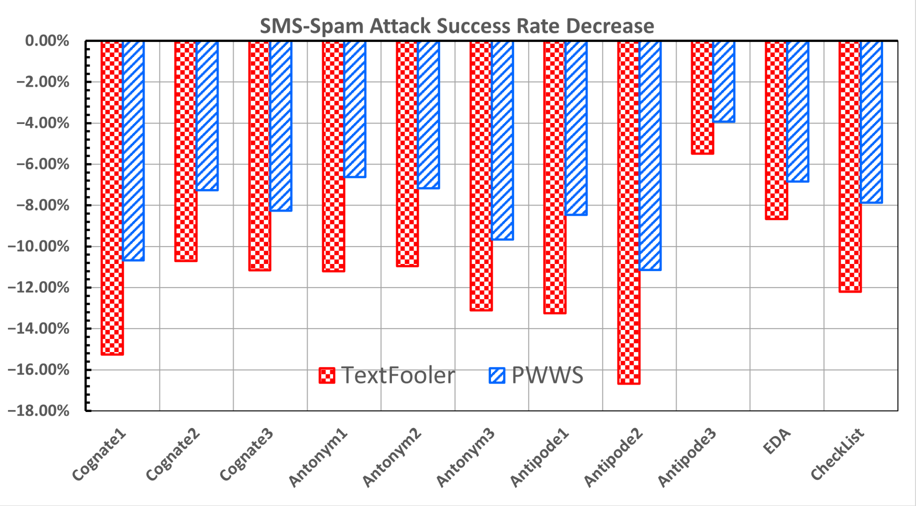


Figure 3.12: SMS Spam Attack Success Rate Decrease.

and semantics into account. JTVAE [28] incorporates a graph message-passing network for molecular graph generation. Despite efforts to develop VAEs for molecular generation, the latent space of VAEs is too small to generate realistic molecules, thereby the molecules generated by VAEs are oversmoothed.

GraphAF [45] employs a flow-based autoregressive model to generate molecular graphs. EDM [32] generates molecules in a 3D combing diffusion process and an equivariant network with both continuous and discrete features. However, the flow-based models require the computationally expensive Jacobian matrix and a reversible network with limited representability, and the diffusion process requires multi-sampling with expensive training costs.

SeqGAN [46] incorporates MCTS-based RL into the GAN architecture for discrete sequence generation. ORGAN [47] specializes SeqGAN [46] by integrating domain-specific knowledge into the reward function. TransORGAN [31] improves the semantic and syntax parsing capabilities with Transformer [40] architecture and variant SMILES. MolGAN [30] uses an implicit, likelihood-free GAN to specialize in the generation of small molecular graphs. Most MCTS-based RL GANs require extensive sampling processes, resulting in high computational costs, and training such GANs on large datasets is challenging. Therefore, in this chapter, we introduced an efficient reward function combined with an LSTM-based GAN for token-level SMILES generation. Our model showed the effectiveness on large datasets with low computational cost.

4.2 EarlGAN

Figure 4.1 shows the overview of EarlGAN. EarlGAN consists of two components: a generator and a discriminator. The LSTM-based generator generates

SMILES autoregressively, and the bi-LSTM-based discriminator predicts the realness of input SMILES at the token-level. The generator update is based on the moment and global rewards, and the maximized entropy. The moment and global rewards are computed from the prediction probabilities of the discriminator, and the maximized entropy is computed from the output probabilities of the tokens of the generator.

4.2.1 Original GAN

The original GAN [41] models the training data distribution using the generator-discriminator competition circle. The value function of the original GAN [41] is described as follows:

$$\begin{aligned}&\min_{\theta}\max_{\phi} V(G_\theta, D_\phi) = \\ &\mathbb{E}_{x\sim p_{data}(x)}[logD_\phi(x)]+ \\ &\mathbb{E}_{z\sim p_z(z)}[log(1-D_\phi(G_\theta(z)))],\end{aligned} \tag{4.1}$$

where G_θ and D_ϕ are the generator and discriminator, θ and ϕ are their parameters, respectively. $p_{data}(x)$ and $p_z(z)$ are the real data and noise distributions, respectively.

4.2.2 Autoregressive GAN

Naively computing the reward based only on the end of the whole sequence can lead to the sparse reward problem [34, 35]. Especially in the early training stage, the discriminator can easily predict the realness of the generated samples, leading to the gradient vanishing. This requires a large number of samplings and is therefore time-consuming. Therefore, we used the assigned prediction probability as a reward for each token. The value function of EarlGAN is as follows:

$$\begin{aligned}&\min_{\theta}\max_{\phi} V(G_\theta, D_\phi) = \\ &\sum_{t=1}^{T}\mathbb{E}_{x_t\sim p_{data}(x_t|\overrightarrow{x}_{1:t-1})}[logD_\phi(y_t|\overrightarrow{x}_{1:t}, \overleftarrow{x}_{t:T})]+ \\ &\sum_{t=1}^{T}\mathbb{E}_{x_t\sim p_\theta(x_t|\overrightarrow{x}_{1:t-1})}[log(1-D_\phi(y_t|\overrightarrow{x}_{1:t}, \overleftarrow{x}_{t:T}))],\end{aligned} \tag{4.2}$$

where y_t is the prediction by the discriminator based on the forward pass $\overrightarrow{x}_{1:t}$ and the backward pass $\overleftarrow{x}_{t:T}$ of the sequence. $p_\theta(x_t|\overrightarrow{x}_{1:t-1})$ is the data distribution generated by the generator and T is the maximum sequence length of the SMILES string.

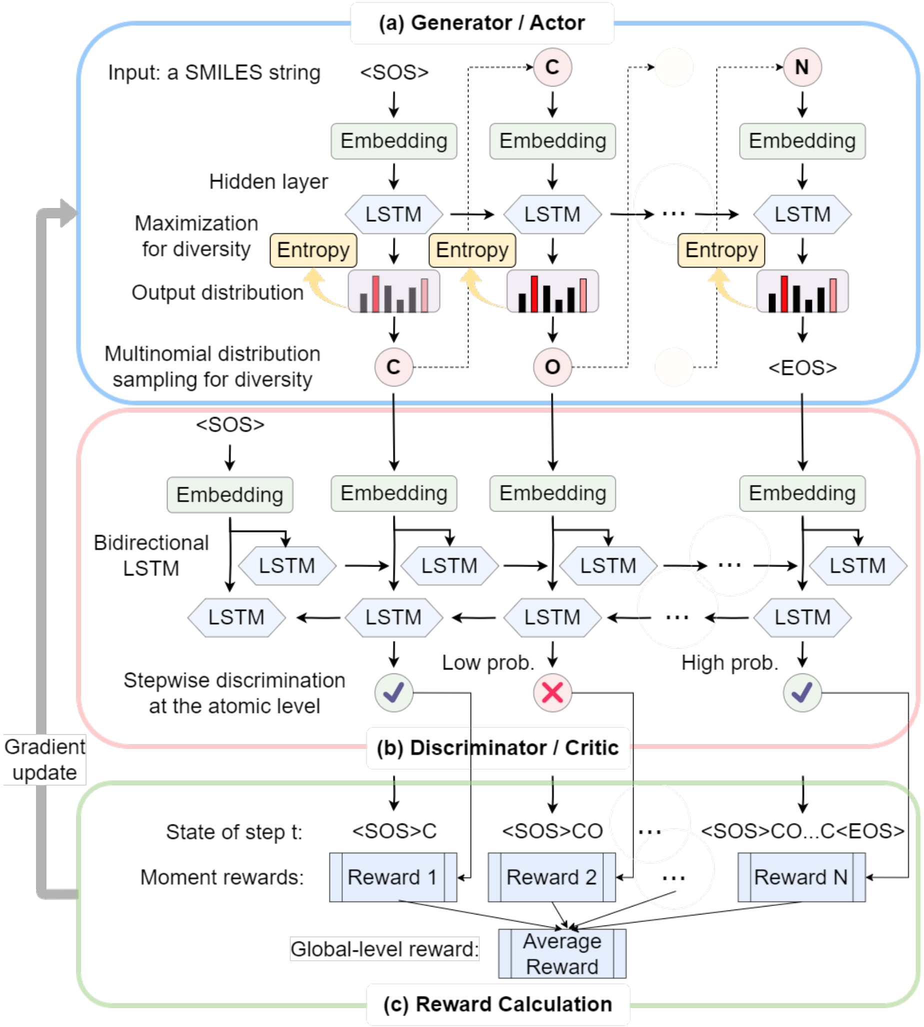


Figure 4.1: Overview of EarlGAN.

4.2.3 Reward Calculation

We used moment and global [38] rewards with information entropy maximization to update the generator instead of the cumulative reward. The combination of moment and global rewards reduces the computational cost and ensures molecular consistency, acting as an alternative to the cumulative reward. Furthermore, the information entropy maximization provides a diverse exploration. The combination of moment and global rewards R_t is described as follows:

$$R_t = 2r_t - 1 + w_{GR} \sum_{t=1}^{T} \frac{r_t}{T}, \tag{4.3}$$

$$r_t = D_\phi(y_t | \overrightarrow{x}_{1:t}, \overleftarrow{x}_{t:T}), \tag{4.4}$$

where $2r_t - 1$ is the moment reward and $\sum_{t=1}^{T} \frac{r_t}{T}$ is the global reward. w_{GR} is the weight assigned to the global reward.

We combined the above reward combination with information entropy maximization. In addition, we used the reward baseline, as in general RL algorithms, to emphasize the relative reward and reduce the sampling variance. We computed the baseline as a global moving average of the mean reward over the entire batch [48]. The final loss function $\mathcal{L}_G$ is described as follows:

$$\mathcal{L}_G = \mathcal{L}_R + w_E \mathcal{L}_E, \tag{4.5}$$

$$\mathcal{L}_R = -\frac{\sum_{n=1}^{N} \sum_{t=1}^{T_n} (R_t^n - b_i) log p_\theta(x_t^n | \overrightarrow{x^n}_{1:t-1})}{\sum_{n=1}^{N} T_n}, \tag{4.6}$$

$$\mathcal{L}_E = -\frac{\sum_{n=1}^{N} \sum_{t=1}^{T_n} H(x_t^n)}{\sum_{n=1}^{N} T_n}, \tag{4.7}$$

$$b_i = \alpha b_{i-1} + (1 - \alpha) R_i, \tag{4.8}$$

$$H(x_t^n) = -\sum_{v=1}^{V} p_\theta(x_t^v) log p_\theta(x_t^v), \tag{4.9}$$

where $\mathcal{L}_R$ and $\mathcal{L}_E$ are the loss functions for the reward combination and entropy, respectively. N is the batch size, b_i is the baseline of the current i-th batch, T_n is the length of the n-th SMILES string in the batch. α is set to 0.9. $H(x_t^n)$ is the Shannon entropy of the t-th atom in the n-th SMILES string, V is the vocabulary size of the dataset, and w_E is the weight assigned to the information entropy.

4.3 Evaluation Experiments

4.3.1 Datasets

We conducted the evaluation experiment, the ablation study, and the visualization analysis on two public datasets: QM9[49] and ZINC [50, 51]. These datasets contain about 130k and 250k molecules, respectively.

4.3.2 Evaluation Metrics

We followed SpotGAN [52] to evaluate EarlGAN using validity, uniqueness, novelty, and diversity. Validity is the ratio of chemically valid SMILES strings to all generated SMILES strings. Uniqueness is the ratio of valid & unique SMILES strings to all valid SMILES strings. Novelty is the ratio of valid & unique SMILES strings that are not present in the training dataset to all valid & unique SMILES strings. Validity, uniqueness, and novelty are defined as follows:

$$Validity = \frac{N_V}{N_G}, \tag{4.10}$$

$$Uniqueness = \frac{N_U}{N_V}, \tag{4.11}$$

$$Novelty = \frac{N_N}{N_U}, \tag{4.12}$$

where N_G is the number of all generated SMILES strings, N_V is the number of generated chemically valid SMILES strings, N_U is the number of generated chemically valid & unique SMILES strings, and N_N is the number of generated chemically valid & unique SMILES strings that are not present in the training dataset.

Diversity represents the distance of chemical structure between molecules based on the Tanimoto coefficient [53] and the Morgan fingerprint [54]. The calculation of the diversity $Div(\mathcal{D}_N)$ between two novel SMILES strings S_i, S_j is as follows:

$$Div(\mathcal{D}_N) = 1 - \frac{1}{|\mathcal{D}_N|} \sum_{M_i, M_j \in \mathcal{D}_N} Sim(M_i, M_j), \tag{4.13}$$

$$Sim(M_i, M_j) = \frac{|M_i \& M_j|}{|M_i| + |M_j| - |M_i \& M_j|}, \tag{4.14}$$

where $Sim(M_i, M_j)$ is the Tanimoto similarity of the Morgan fingerprint M_i, M_j between two arbitrary novel SMILES S_i, S_j. $\mathcal{D}_N$ is the novel SMILES string set, $|\cdot|$ is the number of bits set in the Morgan fingerprint, and & is to count the bits in common between two Morgan fingerprints.

All scores range from 0 to 1. The higher the score, the better the model.

4.3.3 Chemical Properties

We also evaluated the generated valid SMILES using three desirable chemical properties: druglike-ness (QED) [55], solubility (logP), and Synthesizability (SA). Druglike-ness (QED) [55] defines how similar a molecule is to existing drugs. Solubility (logP) defines the hydrophobicity of a molecule and is quantified using the logarithm of the octanol-water partition coefficient. Synthesizability (SA) defines how easy it is to synthesize a molecule using existing chemical reactions based on the synthetic accessibility score [56]. The definition of QED is described as follows:

$$QED = exp(\frac{\sum_{i=1}^{8} W_i ln d_i}{\sum_{i=1}^{8} W_i}), \tag{4.15}$$

where d_i and W_i are the desirability function for the i-th descriptor and the weight assigned to the i-th descriptor, respectively. The eight molecular descriptors are the molecular weight (MW), octanol-water partition coefficient (ALOGP), number of hydrogen bond donors (HBDs), number of hydrogen bond acceptors (HBAs), molecular polar surface area (PSA), number of rotatable bonds (ROTBs), number of aromatic rings (AROMs), and number of structural alerts (ALERTS).

LogP is described as follows:

$$logP = log\frac{c_o}{c_w}, \tag{4.16}$$

where c_o and c_w are the substance activities of the octanol and water phases, respectively.

SA is described as follows:

$$SA = r_s - \sum_{i=1}^{5} p_i, \tag{4.17}$$

where r_s is the experimental knowledge obtained from the molecular synthesis analysis [52]. p_i is five factors: ring complexity, stereo complexity, macrocycle penalty, size penalty, and bridge penalty.

The higher these property scores, the more desirable the generated valid molecules. Druglike-ness (QED) [55] ranges from 0 to 1. Solubility (logP) and Synthesizability (SA) range from 0.1 to 1. The implementation is based on the RDKit tool.

4.3.4 Experiment Settings

We used LSTM as the basic architecture for both the generator and the discriminator, and the latter is based on bi-LSTM. We also employed dropout [57] with a probability of 0.1. For the discriminator, we used the L2 regularization with a coefficient of $1e-6$, and we used the Adam optimizer [58] with a learning rate of $2e-4$. We set the batch size to 1024.

For the QM9[49] dataset, we presented the ablation study and visualization analysis, and for the ZINC [50, 51] dataset, we presented the comparison results, ablation study, and visualization analysis. For the QM9[49] dataset, the maximum training step is set to 20k, and the model was evaluated per 100 steps on 10k samples. For the ZINC [50, 51] dataset, the maximum training step is set to 60k, and the model was evaluated per 100 steps on 5k samples. The maximum string lengths are set to 19 and 49, respectively. The entropy weights w_E were set to $5e-2$ and $1e-3$, respectively. The global reward weights were set to

Table 4.1: Comparison Results with Baselines on ZINC [50, 51] dataset.

Model	Validity(%) ↑	Uniqueness(%) ↑	Novelty(%) ↑	Diversity ↑
Random Sampler [59]	**100.00**	61.54	0.00	0.64
CharacterVAE [29]	73.34	99.18	**100.00**	0.39
GrammarVAE [29]	76.36	**99.55**	**100.00**	0.45
JTVAE [28]	**100.00**	13.94	99.43	0.61
GraphAF^{1*} [45]	72.26	84.80	**100.00**	0.79
GraphAF^{10*} [45]	67.27	99.44	**100.00**	0.69
TransORGAN [31]	75.52	94.64	**100.00**	0.68
EarlGAN [37]	96.14	99.06	99.74	**0.90**

* 1 and 10 represent the minimum length of the SMILES strings.

$5e-3$ and $7.1e-5$, respectively. We repeated the evaluation experiments and the ablation study five times and averaged the results out.

4.3.5 Comparison Results on ZINC Dataset

Table 4.1 shows the comparison results, and the average training time of EarlGAN [37] is 1427.69 minutes. We trained EarlGAN [37] on the entire ZINC [50, 51] dataset. We compared EarlGAN [37] with Random Sampler [59], CharacterVAE [29], GrammarVAE [29], JTVAE [28], GraphAF [45], and TransORGAN [31]. These baseline results were also used in the literature [31]. These baselines were trained on a subset of the ZINC [50, 51] dataset with 5k samples. While EarlGAN did not outperform other baselines in terms of validity, uniqueness, and novelty, its results are comparable to the best of the baselines: validity (96.14%), uniqueness (99.06%), and novelty (99.74%). The diversity of EarlGAN outperformed the other baselines with a value of 0.90. These comparative results indicate that EarlGAN can generate unique & new molecules with diverse structures. We assumed that the balanced performance of EarlGAN is due to the training on a large dataset with an efficient reward function.

4.3.6 Chemical Properties Analysis

We compared valid SMILES and training SMILES in terms of chemical properties on the QM9[49] and ZINC [50, 51] datasets. Table 4.2 shows these comparison results, showing that the mean property values of the generated valid SMILES are quite similar to those of the training SMILES, except for the mean SA on the ZINC [50, 51] dataset. Although the generated valid SMILES have a higher mean SA value than those of the training SMILES, the generated valid SMILES generally align with the training data distribution well.

For a more detailed analysis, we provided some visualization comparisons. Figures 4.2, 4.3 and 4.4 show the violin plots of these property comparisons on the QM9[49] dataset, and Figures 4.5, 4.6 and 4.7 show the violin plots of these property comparisons on the ZINC [50, 51] dataset. Although the distributions of the generated valid SMILES are not identical to those of the training SMILES,

Table 4.2: Chemical Properties Comparison on QM9[49] and ZINC [50, 51].

Data	Max Len	Min Len	Mean Len	Mean QED ↑	Mean LogP ↑	Mean SA ↑
QM9 Real	29	1	15	0.47	0.30	**0.27**
QM9 Valid	19	8	15	**0.49**	**0.34**	0.23
ZINC Real	109	9	44	0.73	**0.56**	0.56
ZINC Valid	49	9	35	**0.76**	0.54	**0.67**

they are similar, supporting the numerical results. The similarity proved that EarlGAN can learn the property distribution of the training data well.

Figures A.1-A.12 show the generated valid and training SMILES with the top 12 property values in terms of drug-likeness (QED), solubility (logP), and SA. These figures proved that EarlGAN generates novel SMILES that are not present in the training datasets.

4.3.7 Representation Visualization on ZINC Dataset

We also visualized the representation of the generated valid and training SMILES on the ZINC [50, 51] dataset using the trained discriminator. Specifically, we first sampled the same number of valid and training SMILES and fed them into the trained discriminator to extract the hidden state of the discriminator. Then, we used the Principal Component Analysis (PCA) to reduce the dimensions of the representations to 2D and 3D for visualization, respectively. Figures 4.8 and 4.9 show the 2D and 3D scatterplots, respectively. The distributions of both datasets are quite similar in both 2D and 3D scatter plots, supporting the idea that EarlGAN learns the training distribution at the representation level. Furthermore, there are some out-of-distribution generated samples, proving that EarlGAN does not merely memorize the training data.

4.3.8 Ablation Study

Table 5.1 shows the ablation study results on the QM9[49] and ZINC [50, 51] datasets. We removed both information entropy maximization and global reward as a comparison, and only global reward as a further comparison. The training processes were also repeated five times. For the QM9[49] dataset, information entropy maximization increased the validity, uniqueness, and novelty, especially novelty, with increased computational cost. The global reward mainly increased the uniqueness. For the ZINC [50, 51] dataset, the information entropy maximization increased the validity with increased computational cost, and the global reward increased all metrics.

Although the effectiveness of information entropy maximization may vary depending on the dataset, it provided a more diverse exploration, resulting in a better model parameter to improve validity. The global reward mainly improved the uniqueness. The improved validity on the ZINC [50, 51] dataset due to the global reward on the ZINC dataset is straightforward: the improved molecular consistency increased the stability of molecular generation. However, the

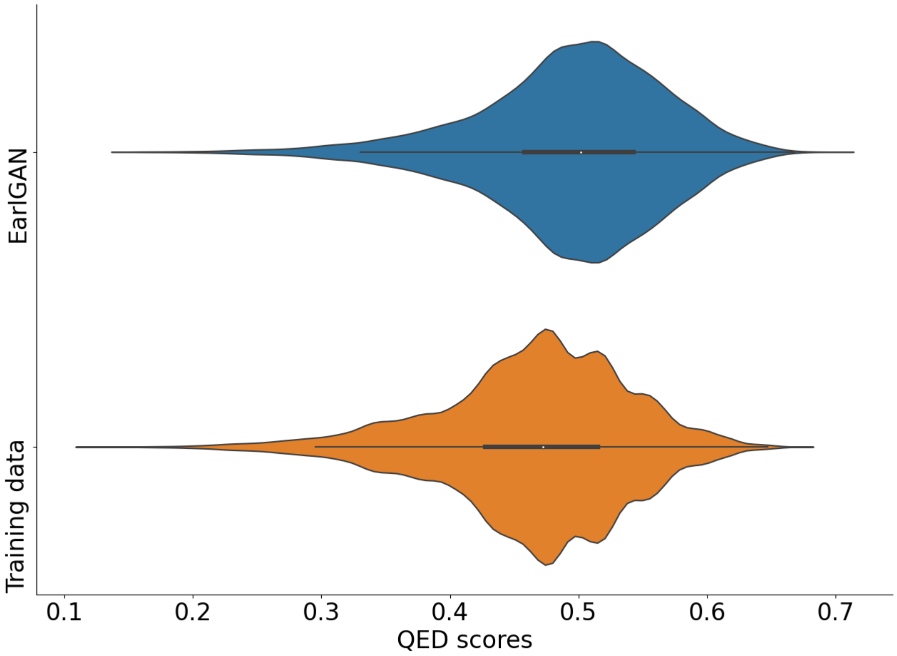


Figure 4.2: QED Violin Plot Comparison of Generated Valid (upper) and Training (lower) SMILES on QM9.

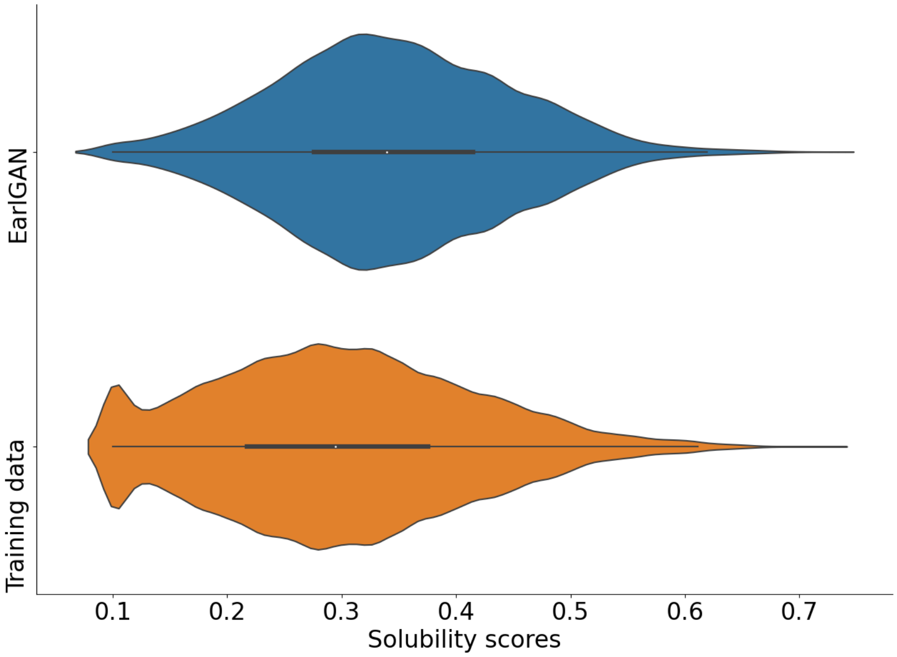


Figure 4.3: Solubility Violin Plot Comparison of Generated Valid (upper) and Training (lower) SMILES on QM9.

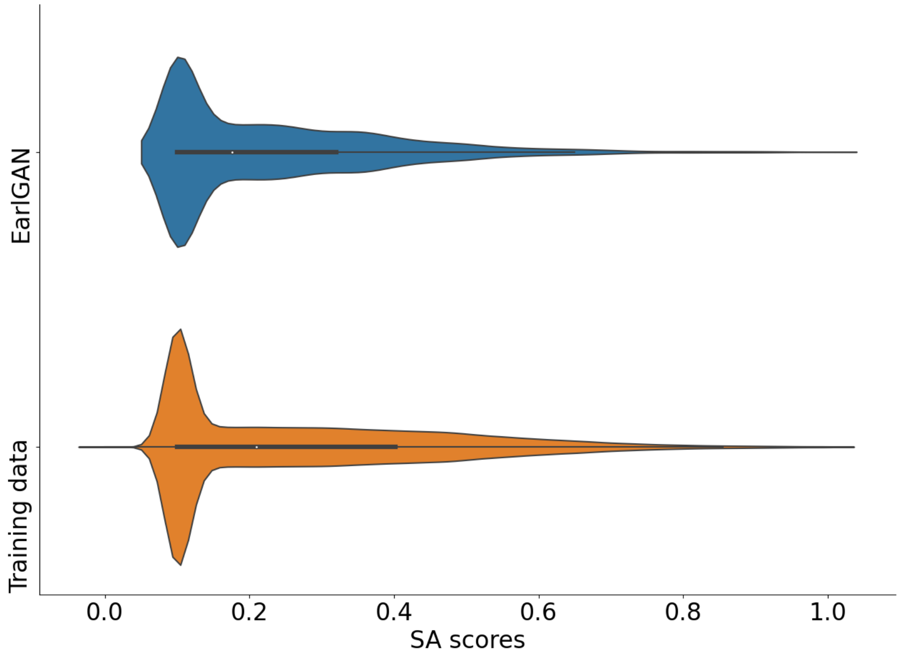


Figure 4.4: SA Violin Plot Comparison of Generated Valid (upper) and Training (lower) SMILES on QM9.

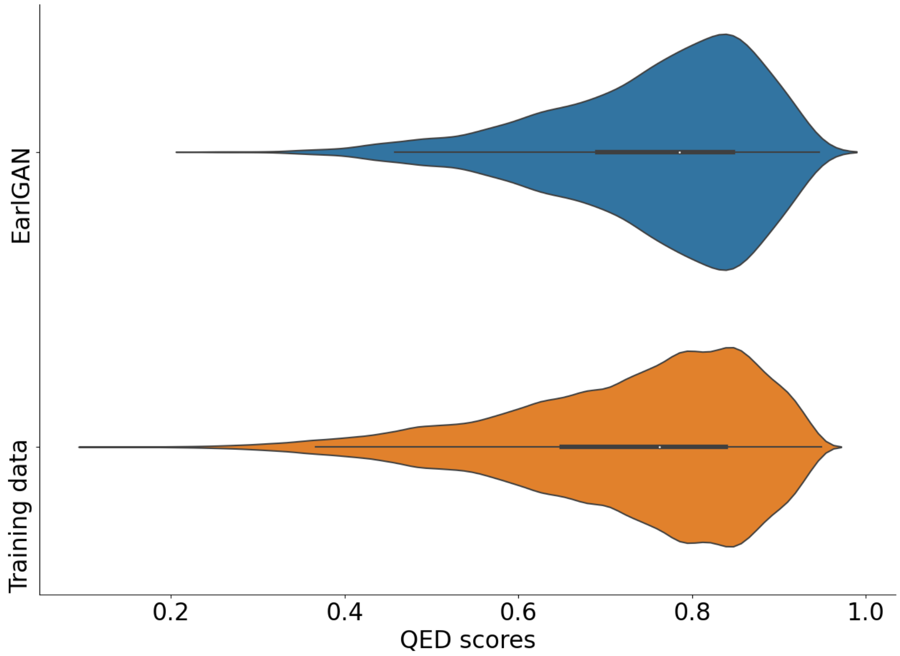


Figure 4.5: QED Violin Plot Comparison of Generated Valid (upper) and Training (lower) SMILES on ZINC.

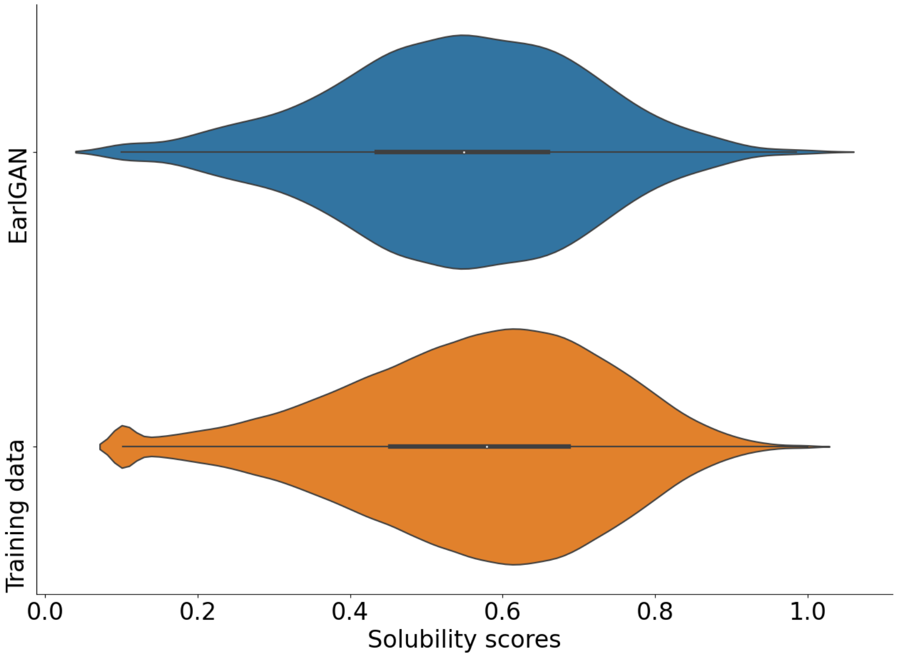


Figure 4.6: Solubility Violin Plot Comparison of Generated Valid (upper) and Training (lower) SMILES on ZINC.

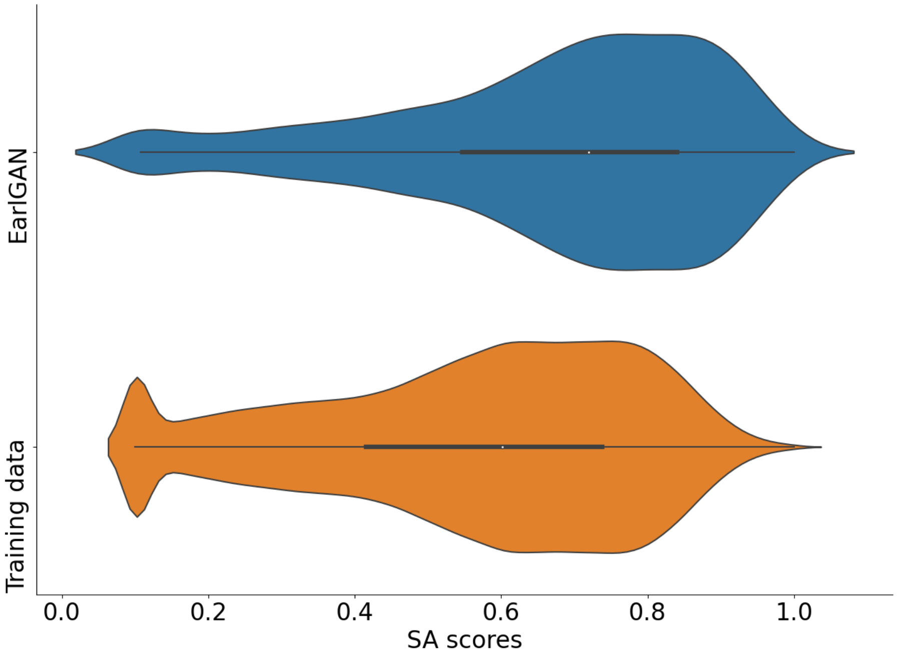


Figure 4.7: SA Violin Plot Comparison of Generated Valid (upper) and Training (lower) SMILES on ZINC.

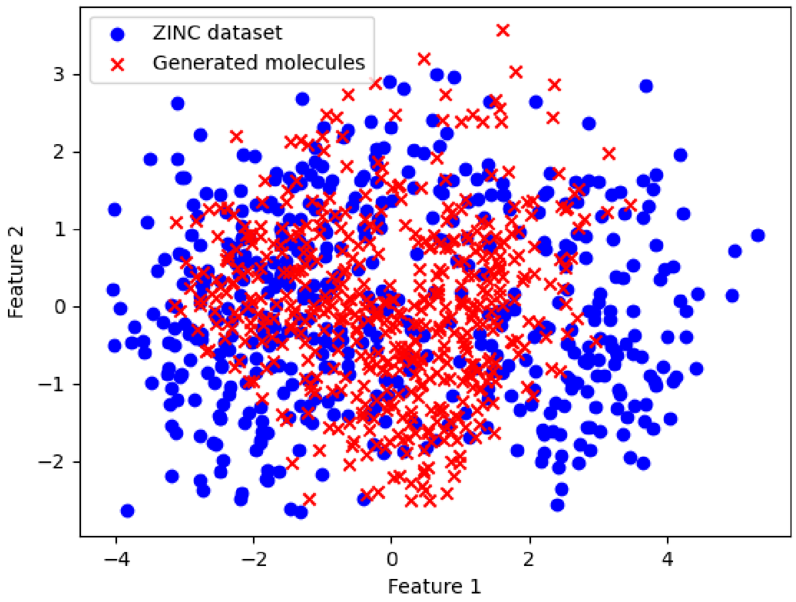


Figure 4.8: Representation Visualization in 2D Scatter Plot on ZINC.

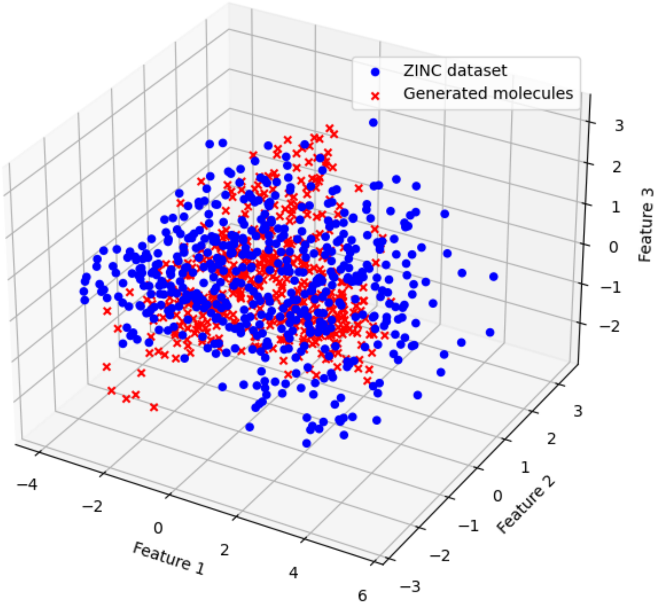


Figure 4.9: Representation Visualization in 3D Scatter Plot on ZINC.

Table 5.1: Ablation Study Results on QM9[49] and ZINC [50, 51] Datasets.

Dataset		Validity(%) ↑	Uniqueness(%) ↑	Novelty(%) ↑	Diversity ↑	Time(min) ↓
QM9	w/o E and GR	94.41	83.27	65.55	0.92	**178.20**
	w/o GR	**94.45**	84.35	**70.39**	0.92	192.53
	EarlGAN	94.07	**86.24**	70.04	0.92	196.25
ZINC	w/o E and GR	95.26	98.88	**99.79**	0.90	**1368.38**
	w/o GR	95.87	98.54	99.67	0.90	1468.35
	EarlGAN	**96.14**	**99.06**	99.74	0.90	1427.69

w/o means without the specific components, E and GR mean the entropy maximization and global reward component, respectively.

improved uniqueness is surprising. The combination of these two components contributed comprehensively to the model.

5 Conclusion

In this thesis, we studied sequence modeling for text data augmentation and molecular generation.

In Chapter 3, we introduced three text data augmentation methods to improve the robustness of LMs. We conducted extensive evaluation experiments that validated the effectiveness of our methods. In particular, the Antonym-based and Antipode-based methods generally improved the robustness because more out-of-distribution samples can be learned.

In Chapter 4, we introduced EarlGAN with an efficient reward function instead of the cumulative reward. The efficiency of the reward function makes EarlGAN suitable for large molecular datasets. EarlGAN showed a balanced trade-off between validity, uniqueness, and novelty. The visualization analysis and ablation study provided insight into the generated valid SMILES. EarlGAN provides an alternative for training on large molecular datasets.

References

[1] Hossam Faris, Al-Zoubi Ala'M, Ali Asghar Heidari, Ibrahim Aljarah, Majdi Mafarja, Mohammad A Hassonah, and Hamido Fujita. An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. *Information Fusion*, 48:67–83, 2019.

[2] Kheir Eddine Daouadi, Rim Zghal Rebaï, and Ikram Amous. Optimizing semantic deep forest for tweet topic classification. *Information Systems*, 101:101801, 2021.

[3] Feifan Fan, Yansong Feng, and Dongyan Zhao. Multi-grained attention network for aspect-level sentiment classification. In *Proceedings of the 2018*

Conference on Empirical Methods in Natural Language Processing, pages 3433–3442, 2018.

[4] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8018–8025, 2020.

[5] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online, July 2020. Association for Computational Linguistics.

[6] Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5053–5069, Online, June 2021. Association for Computational Linguistics.

[7] Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy, July 2019. Association for Computational Linguistics.

[8] Jason Wei and Kai Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China, November 2019. Association for Computational Linguistics.

[9] Akbar Karimi, Leonardo Rossi, and Andrea Prati. AEDA: An easier data augmentation technique for text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2748–2754, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[10] Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. Data boost: Text data augmentation through reinforcement learning guided conditional generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9031–9041, Online, November 2020. Association for Computational Linguistics.

[11] Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. Do

not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390, 2020.

[12] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268, 2020.

[13] Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[14] Gözde Gül Şahin and Mark Steedman. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5004–5009, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[15] Tong Niu and Mohit Bansal. Automatically learning data augmentation policies for dialogue tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1317–1323, Hong Kong, China, November 2019. Association for Computational Linguistics.

[16] Huidong Tang, Sayaka Kamei, and Yasuhiko Morimoto. Data augmentation methods for enhancing robustness in text classification tasks. *Algorithms*, 16(1):59, 2023.

[17] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 28, 2015.

[18] Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics*, 2002.

[19] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*, 2001.

[20] Alexis Conneau and Douwe Kiela. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA).

[21] Tiago A Almeida, José María G Hidalgo, and Akebo Yamakami. Contributions to the study of sms spam filtering: new collection and results. In *Proceedings of the 11th ACM Symposium on Document Engineering*, pages 259–262, 2011.

[22] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv Preprint ArXiv:1910.01108*, 2019.

[23] Nurken Berdigaliyev and Mohamad Aljofan. An overview of drug discovery and development. *Future Medicinal Chemistry*, 12(10):939–947, 2020.

[24] Kit-Kay Mak and Mallikarjuna Rao Pichika. Artificial intelligence in drug development: present status and future prospects. *Drug Discovery Today*, 24(3):773–780, 2019.

[25] OpenAI. Gpt-4 technical report, 2023.

[26] James Betker, Gabriel Goh, and et al. Improving image generation with better captions. 2023.

[27] Hanjun Dai, Yingtao Tian, Bo Dai, Steven Skiena, and Le Song. Syntax-directed variational autoencoder for molecule generation. In *Proceedings of the International Conference on Learning Representations*, 2018.

[28] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *Proceedings of the International Conference on Machine Learning*, pages 2323–2332. PMLR, 2018.

[29] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *Proceedings of the International Conference on Machine Learning*, pages 1945–1954. PMLR, 2017.

[30] Nicola De Cao and Thomas Kipf. Molgan: An implicit generative model for small molecular graphs. *ArXiv Preprint ArXiv:1805.11973*, 2018.

[31] Chen Li, Chikashige Yamanaka, Kazuma Kaitoh, and Yoshihiro Yamanishi. 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*, pages 3884–3890.

[32] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *Proceedings of the International Conference on Machine Learning*, pages 8867–8887. PMLR, 2022.

[33] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988.

[34] Cyprien de Masson d'Autume, Shakir Mohamed, Mihaela Rosca, and Jack Rae. Training language gans from scratch. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 32, 2019.

[35] William Fedus, Ian Goodfellow, and Andrew M Dai. Maskgan: better text generation via filling in the_. *ArXiv Preprint ArXiv:1801.07736*, 2018.

[36] Huidong Tang, Chen Li, Shuai Jiang, Huachong Yu, Sayaka Kamei, Yoshihiro Yamanishi, and Yasuhiko Morimoto. Macgan: A moment-actor-critic reinforcement learning-based generative adversarial network for molecular generation. In *Proceedings of the 7th APWeb-WAIM International Joint Conference on Web and Big Data*, 2023.

[37] Huidong Tang, Chen Li, Shuai Jiang, Huachong Yu, Sayaka Kamei, Yoshihiro Yamanishi, and Yasuhiko Morimoto. Earlgan: An enhanced actor–critic reinforcement learning agent-driven gan for de novo drug design. *Pattern Recognition Letters*, 175:45–51, 2023.

[38] Jingjing Xu, Xuancheng Ren, Junyang Lin, and Xu Sun. Diversity-promoting gan: A cross-entropy based generative adversarial network for diversified text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3940–3949, 2018.

[39] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv Preprint ArXiv:1810.04805*, 2018.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 30, 2017.

[41] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 27, 2014.

[42] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[43] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[44] Erik Cambria, Yang Li, Frank Z Xing, Soujanya Poria, and Kenneth Kwok. Senticnet 6: Ensemble application of symbolic and subsymbolic ai for sentiment analysis. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 105–114, 2020.

[45] Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. *ArXiv Preprint ArXiv:2001.09382*, 2020.

[46] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[47] Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-reinforced generative adversarial networks (organ) for sequence generation models. *ArXiv Preprint ArXiv:1705.10843*, 2017.

[48] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[49] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1(1):1–7, 2014.

[50] John J Irwin, Teague Sterling, Michael M Mysinger, Erin S Bolstad, and Ryan G Coleman. Zinc: a free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52(7):1757–1768, 2012.

[51] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268–276, 2018.

[52] Chen Li and Yoshihiro Yamanishi. Spotgan: A reverse-transformer gan generates scaffold-constrained molecules with property optimization. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 323–338. Springer, 2023.

[53] Taffee T Tanimoto. Elementary mathematical theory of classification and prediction. 1958.

[54] Adrià Cereto-Massagué, María José Ojeda, Cristina Valls, Miquel Mulero, Santiago Garcia-Vallvé, and Gerard Pujadas. Molecular fingerprint similarity search in virtual screening. *Methods*, 71:58–63, 2015.

[55] G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature Chemistry*, 4(2):90–98, 2012.

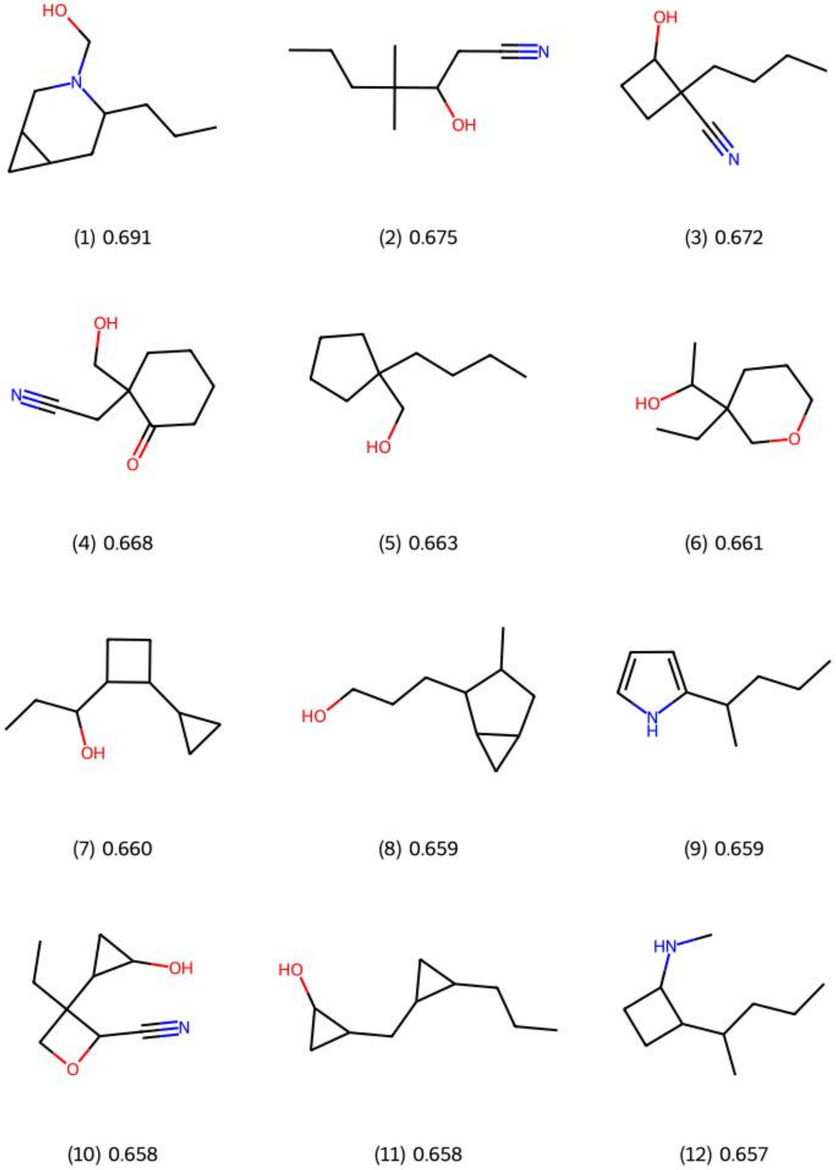
[56] Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of Cheminformatics*, 1:1–11, 2009.

[57] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[58] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ArXiv Preprint ArXiv:1412.6980*, 2014.

[59] Wenhao Gao and Connor W Coley. The synthesizability of molecules proposed by generative models. *Journal of Chemical Information and Modeling*, 60(12):5714–5723, 2020.

A Appendix



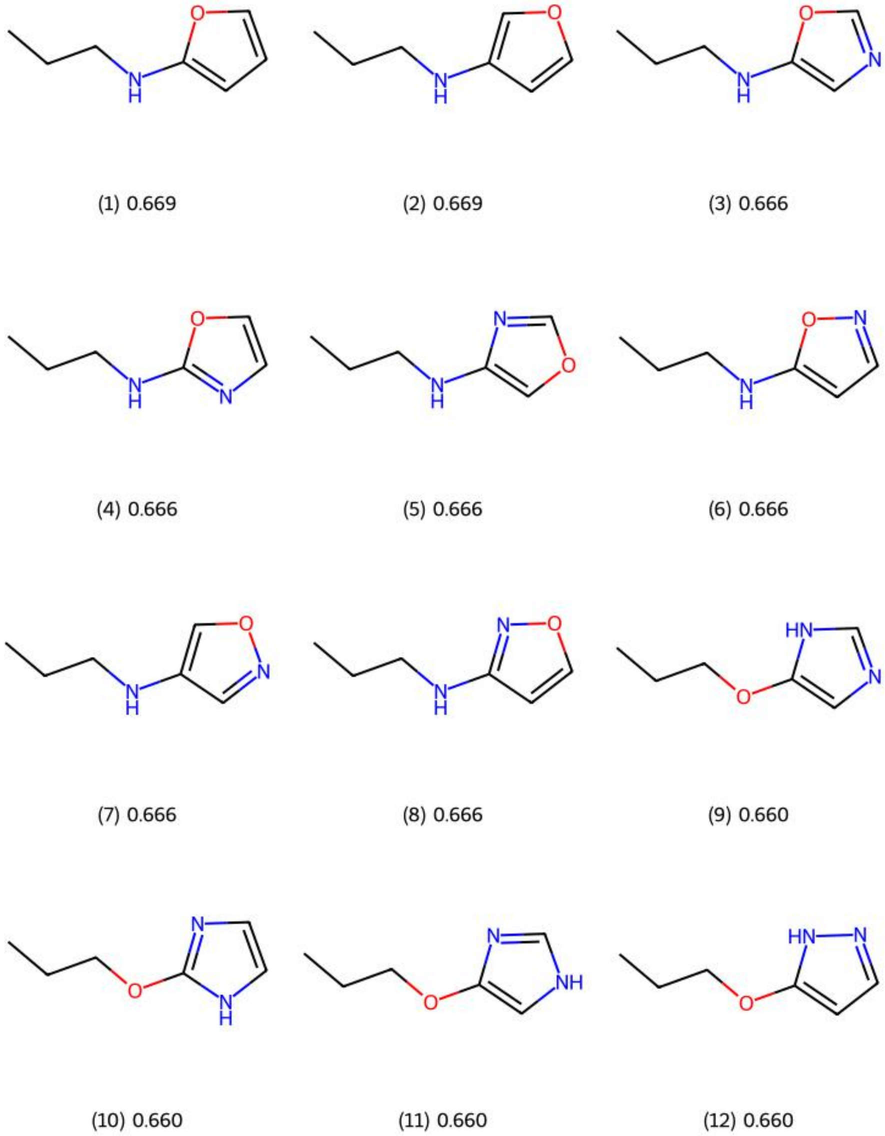
(1) 0.691 (2) 0.675 (3) 0.672

(4) 0.668 (5) 0.663 (6) 0.661

(7) 0.660 (8) 0.659 (9) 0.659

(10) 0.658 (11) 0.658 (12) 0.657

Figure A.1: Top 12 Drug-likeness (QED) Valid Molecules on QM9.



(1) 0.669

(2) 0.669

(3) 0.666

(4) 0.666

(5) 0.666

(6) 0.666

(7) 0.666

(8) 0.666

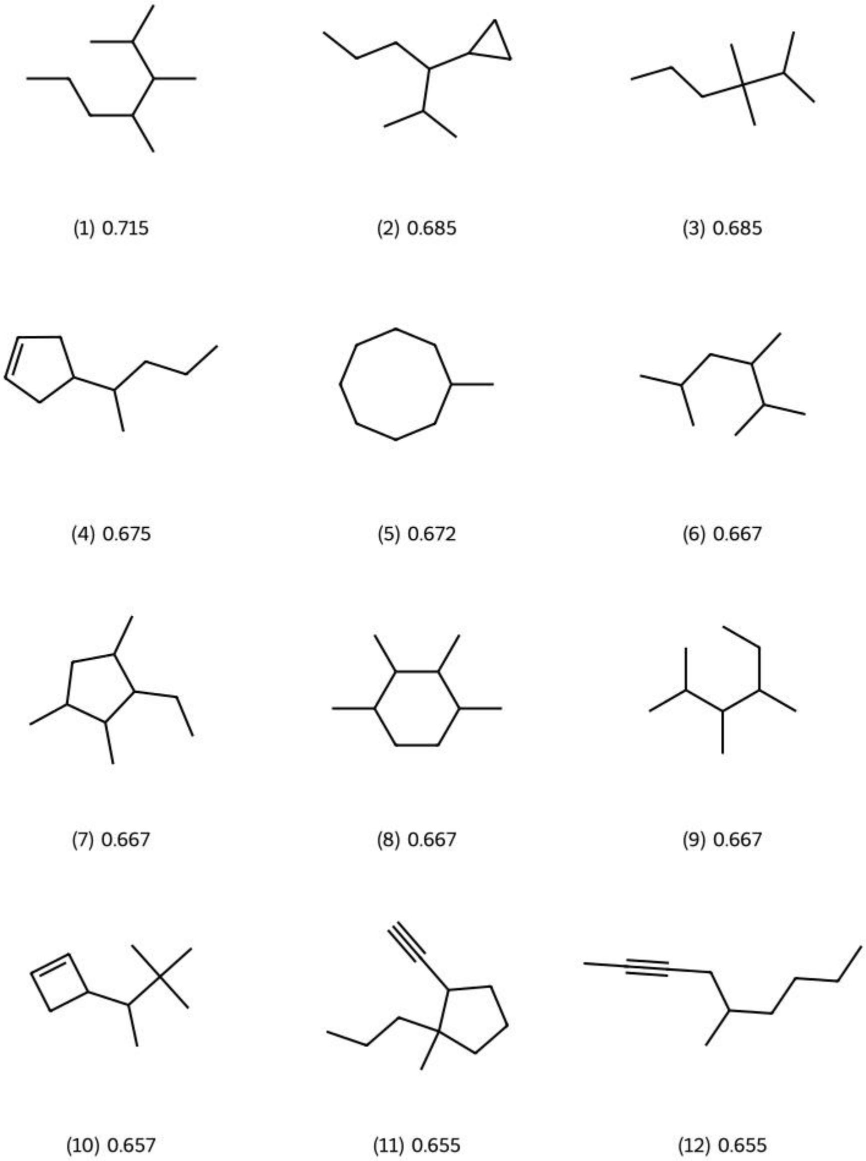
(9) 0.660

(10) 0.660

(11) 0.660

(12) 0.660

Figure A.2: Top 12 Drug-likeness (QED) Real Molecules on QM9.



(1) 0.715

(2) 0.685

(3) 0.685

(4) 0.675

(5) 0.672

(6) 0.667

(7) 0.667

(8) 0.667

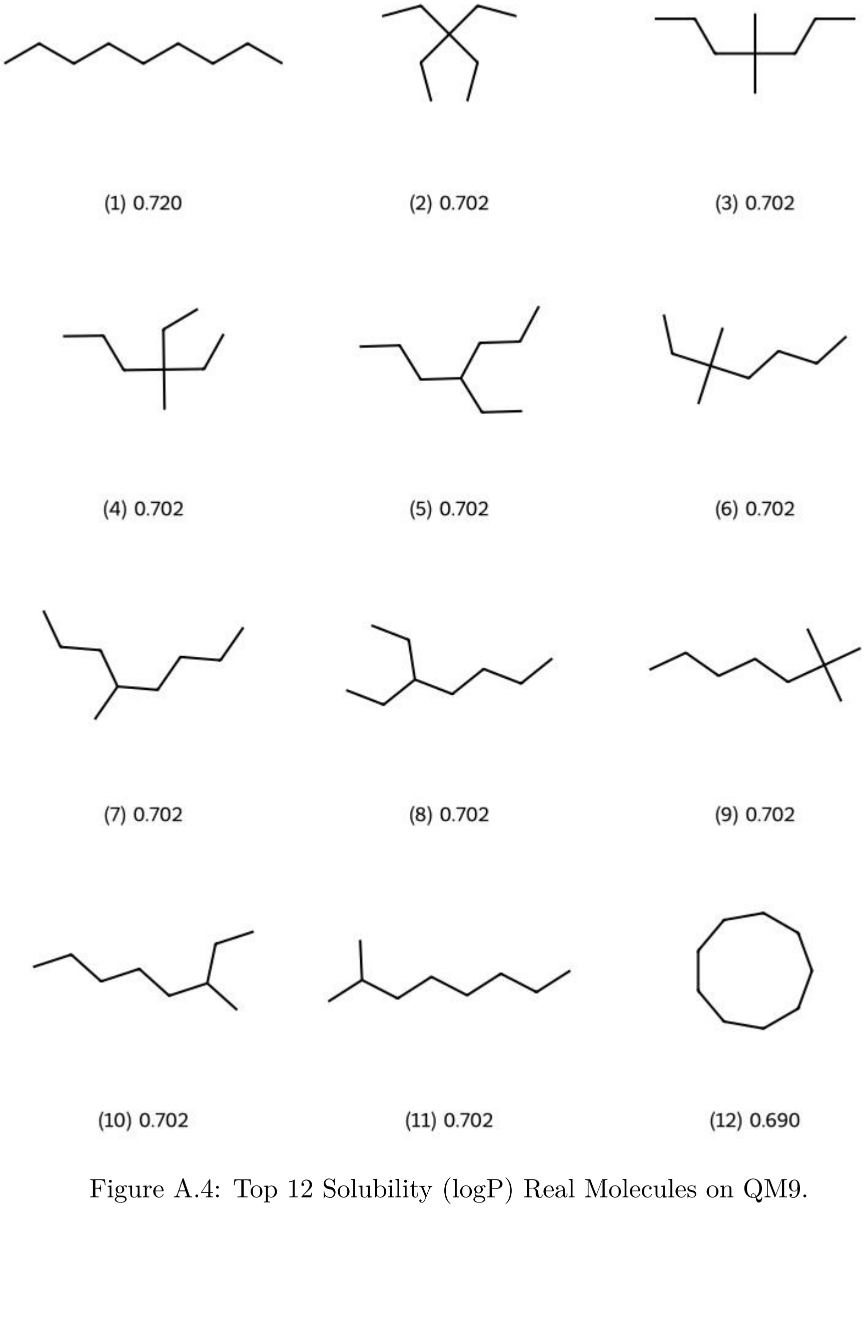
(9) 0.667

(10) 0.657

(11) 0.655

(12) 0.655

Figure A.3: Top 12 Solubility (logP) Valid Molecules on QM9.



(1) 0.720

(2) 0.702

(3) 0.702

(4) 0.702

(5) 0.702

(6) 0.702

(7) 0.702

(8) 0.702

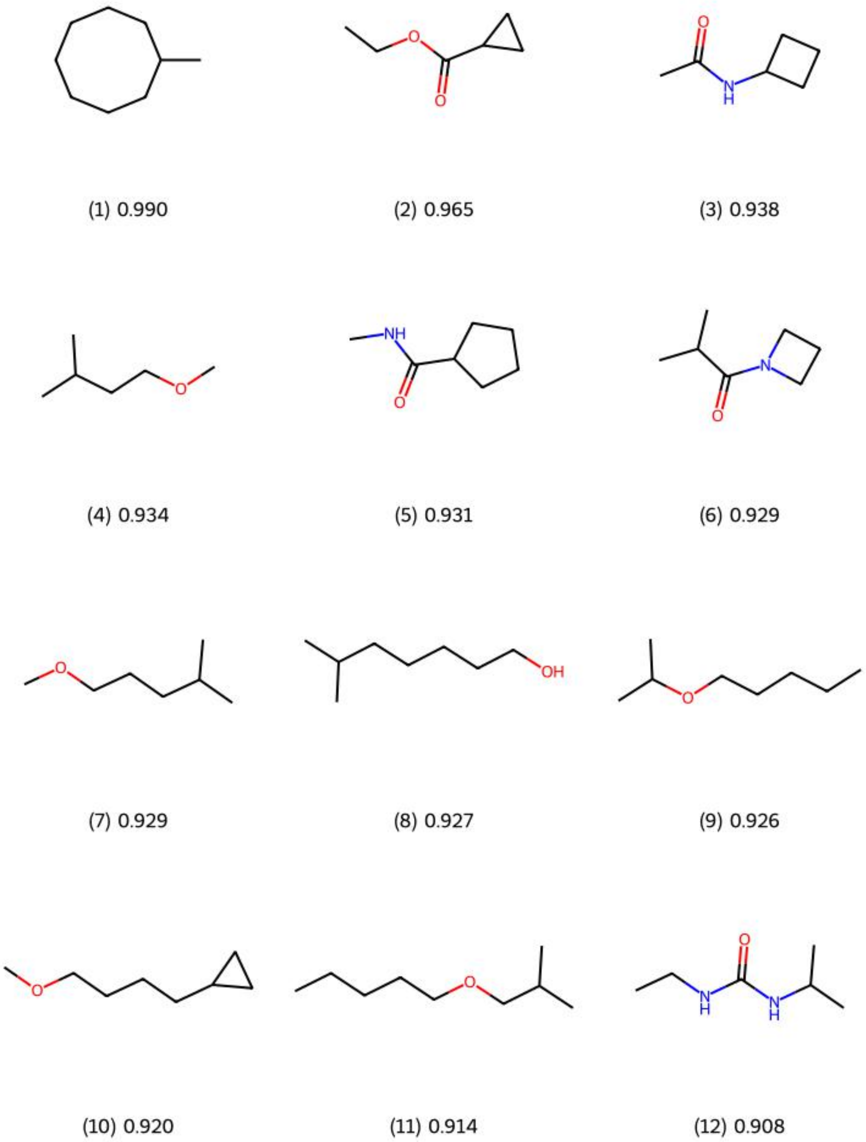
(9) 0.702

(10) 0.702

(11) 0.702

(12) 0.690

Figure A.4: Top 12 Solubility (logP) Real Molecules on QM9.



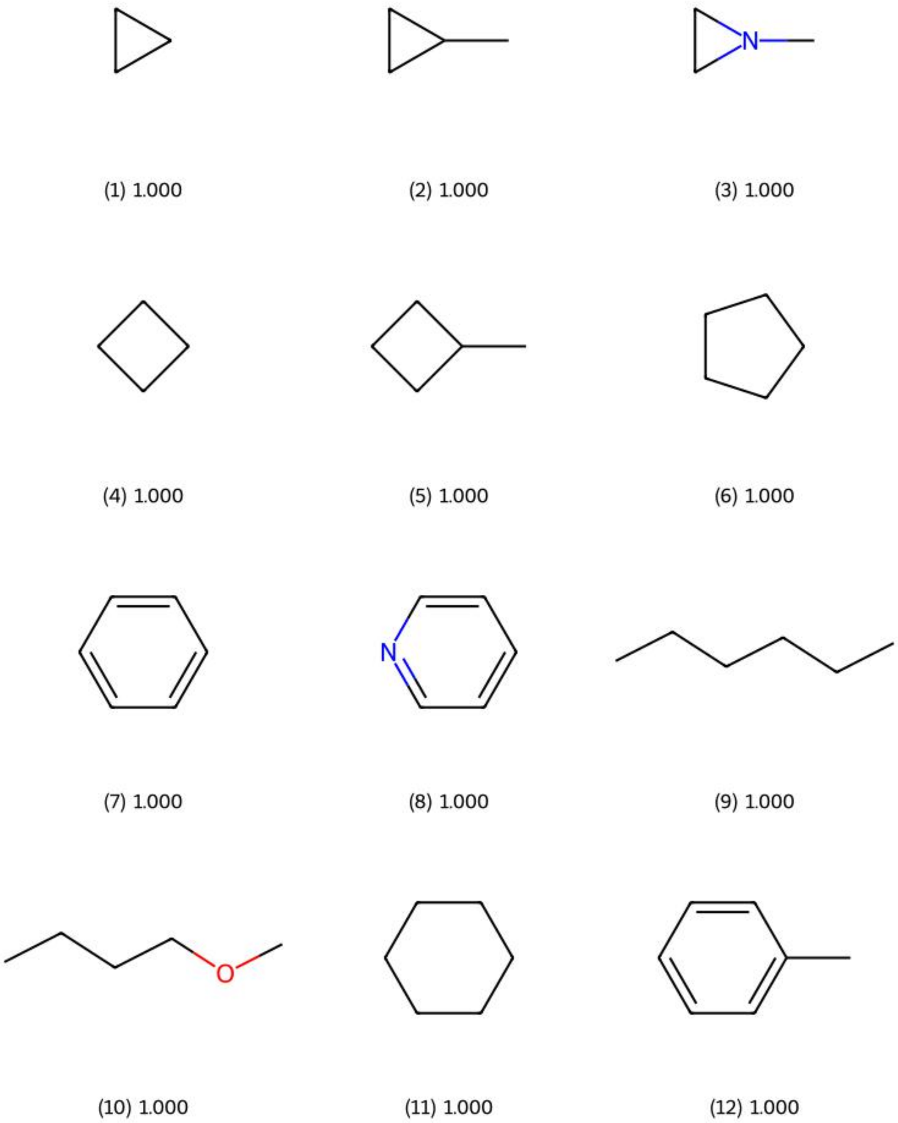
(1) 0.990 (2) 0.965 (3) 0.938

(4) 0.934 (5) 0.931 (6) 0.929

(7) 0.929 (8) 0.927 (9) 0.926

(10) 0.920 (11) 0.914 (12) 0.908

Figure A.5: Top 12 SA Valid Molecules on QM9.



(1) 1.000

(2) 1.000

(3) 1.000

(4) 1.000

(5) 1.000

(6) 1.000

(7) 1.000

(8) 1.000

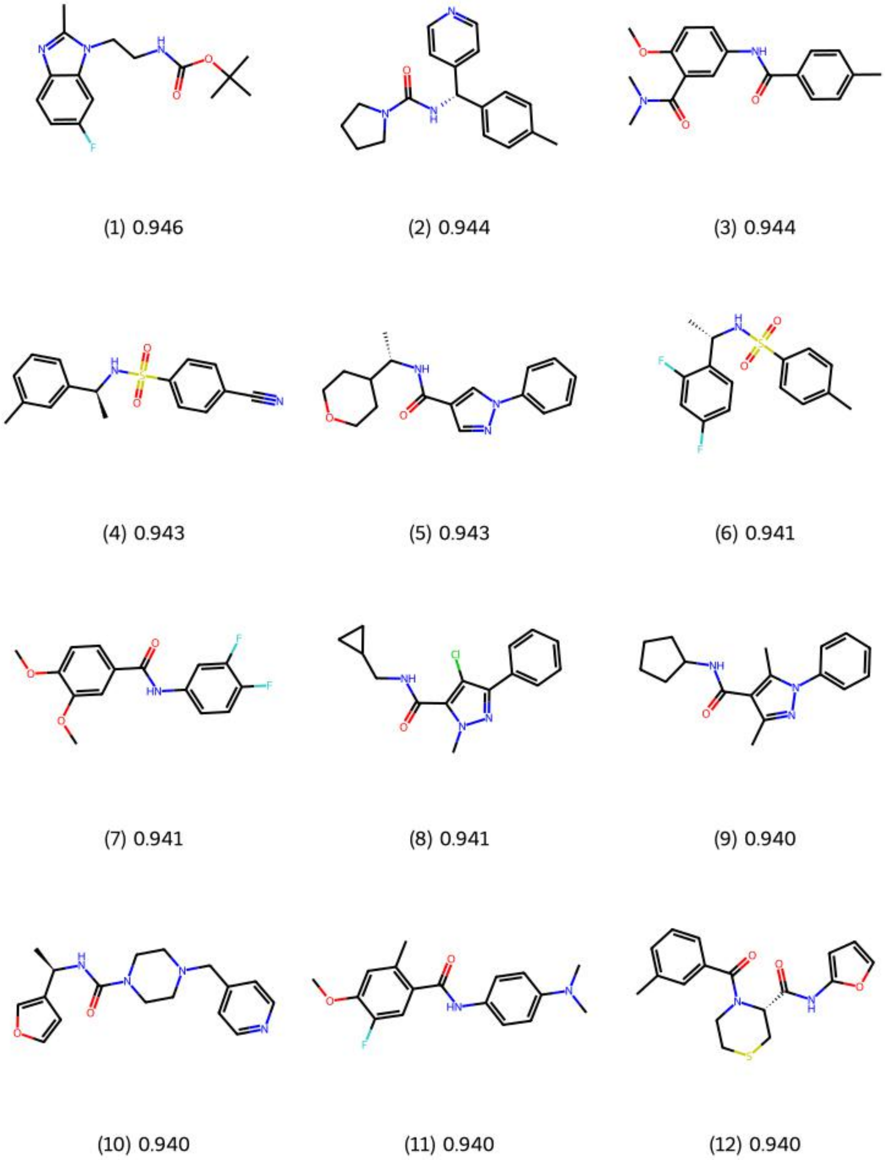
(9) 1.000

(10) 1.000

(11) 1.000

(12) 1.000

Figure A.6: Top 12 SA Real Molecules on QM9.



(1) 0.946

(2) 0.944

(3) 0.944

(4) 0.943

(5) 0.943

(6) 0.941

(7) 0.941

(8) 0.941

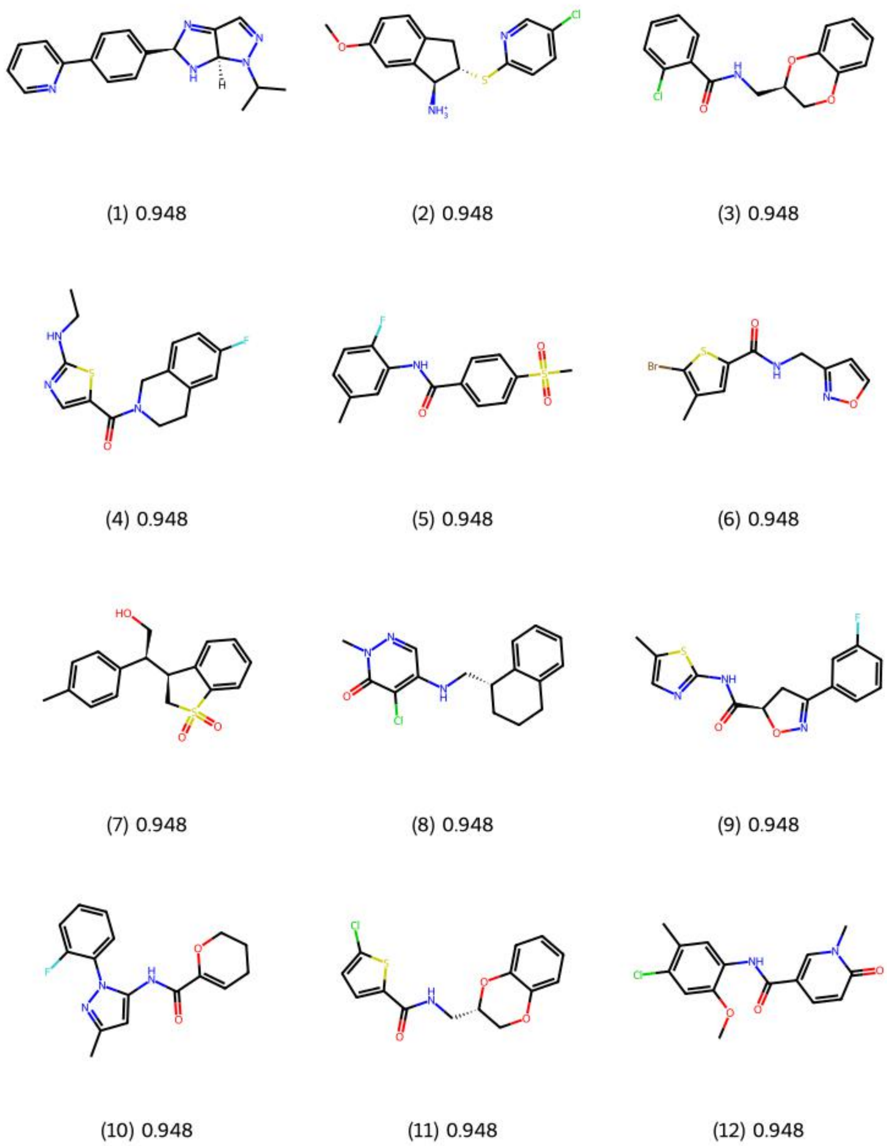
(9) 0.940

(10) 0.940

(11) 0.940

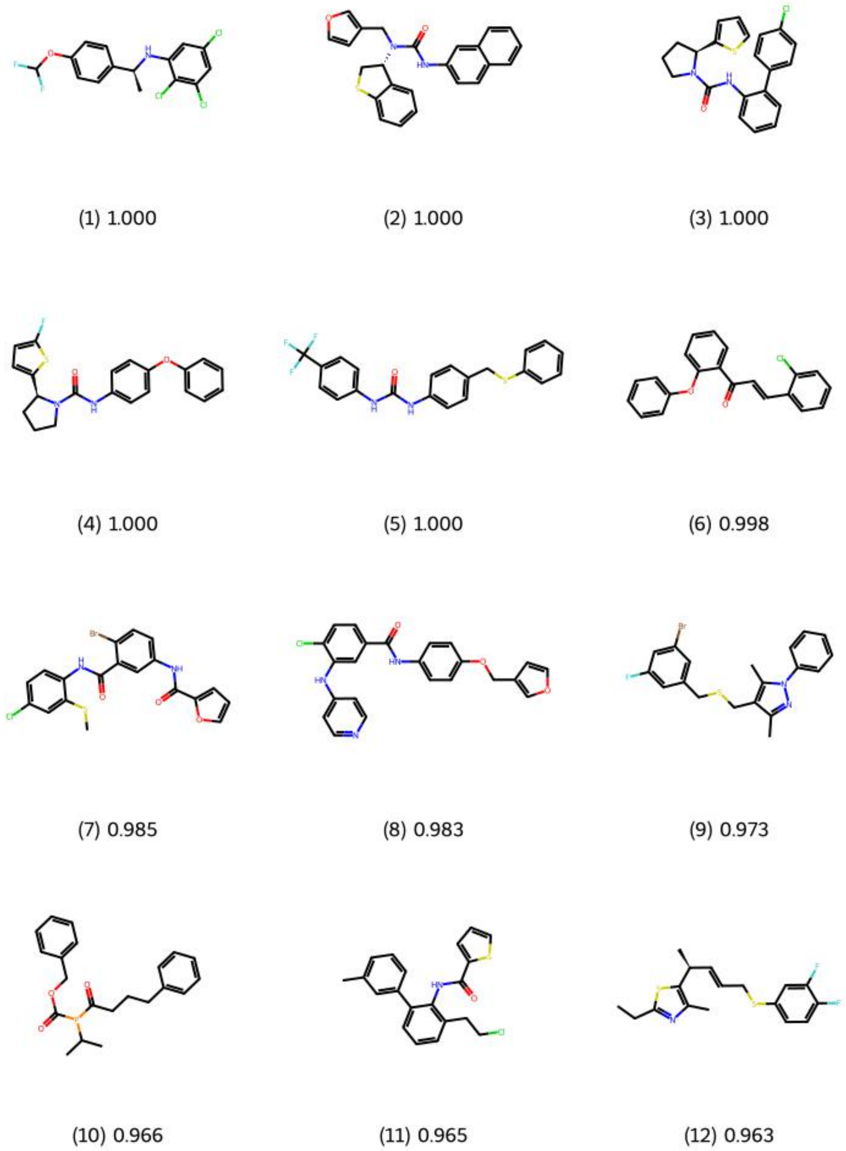
(12) 0.940

Figure A.7: Top 12 Drug-likeness (QED) Valid Molecules on ZINC.



(1) 0.948
(2) 0.948
(3) 0.948
(4) 0.948
(5) 0.948
(6) 0.948
(7) 0.948
(8) 0.948
(9) 0.948
(10) 0.948
(11) 0.948
(12) 0.948

Figure A.8: Top 12 Drug-likeness (QED) Real Molecules on ZINC.



(1) 1.000

(2) 1.000

(3) 1.000

(4) 1.000

(5) 1.000

(6) 0.998

(7) 0.985

(8) 0.983

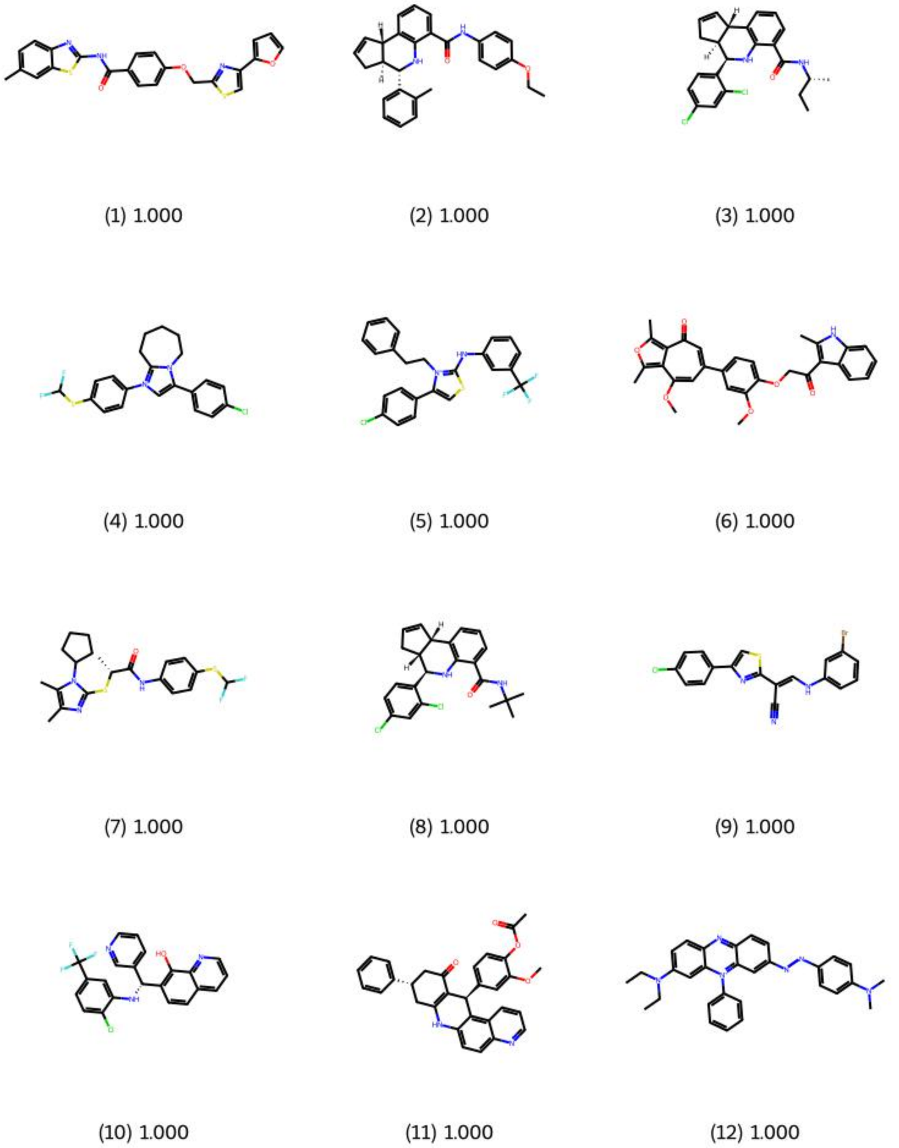
(9) 0.973

(10) 0.966

(11) 0.965

(12) 0.963

Figure A.9: Top 12 Solubility (logP) Valid Molecules on ZINC.



(1) 1.000

(2) 1.000

(3) 1.000

(4) 1.000

(5) 1.000

(6) 1.000

(7) 1.000

(8) 1.000

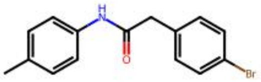
(9) 1.000

(10) 1.000

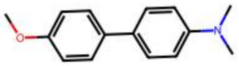
(11) 1.000

(12) 1.000

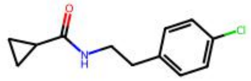
Figure A.10: Top 12 Solubility (logP) Real Molecules on ZINC.



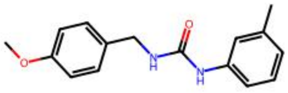
(1) 1.000



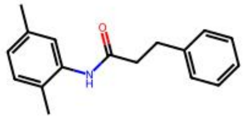
(2) 1.000



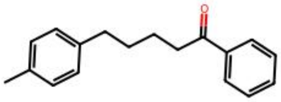
(3) 1.000



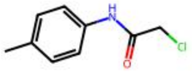
(4) 1.000



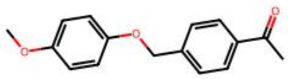
(5) 1.000



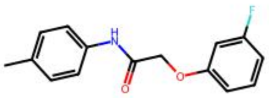
(6) 1.000



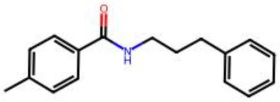
(7) 1.000



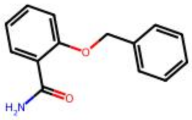
(8) 1.000



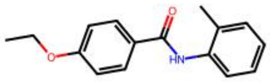
(9) 1.000



(10) 1.000

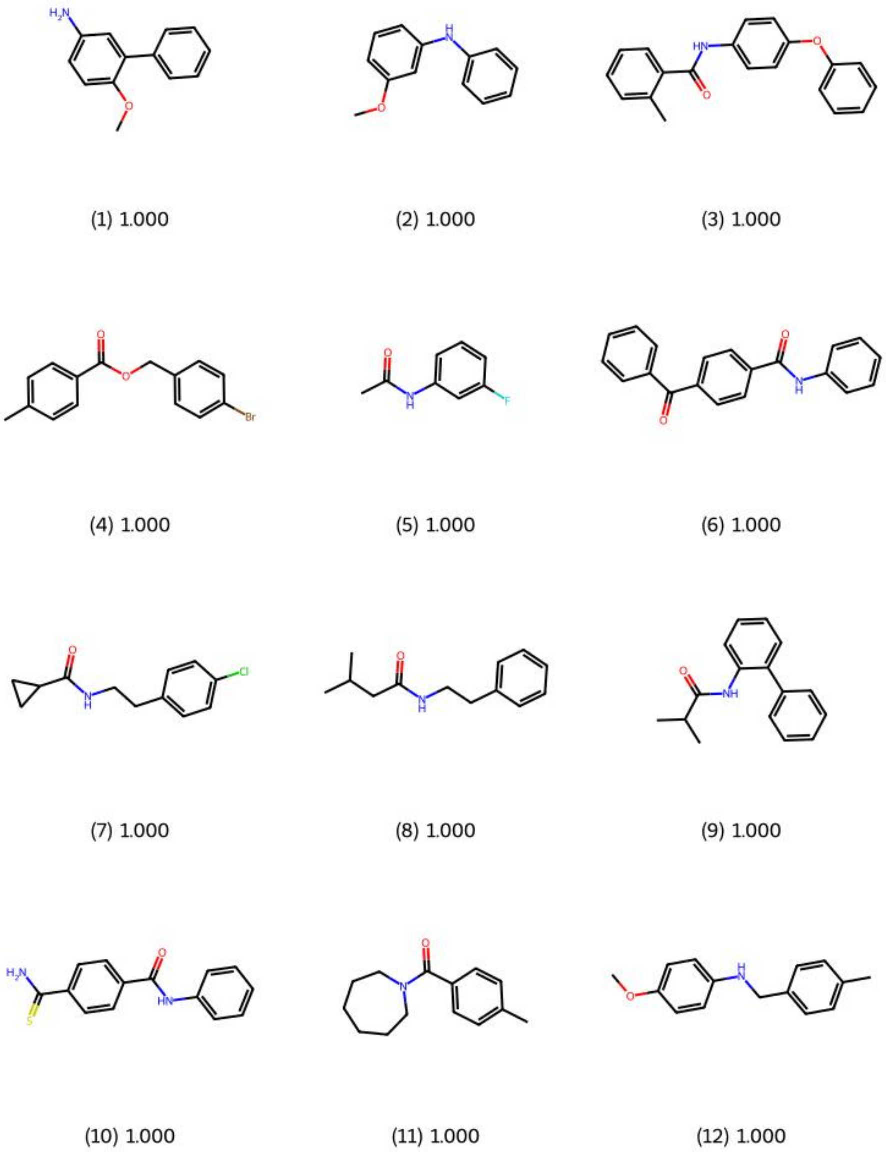


(11) 1.000



(12) 1.000

Figure A.11: Top 12 SA Valid Molecules on ZINC.



(1) 1.000

(2) 1.000

(3) 1.000

(4) 1.000

(5) 1.000

(6) 1.000

(7) 1.000

(8) 1.000

(9) 1.000

(10) 1.000

(11) 1.000

(12) 1.000

Figure A.12: Top 12 SA Real Molecules on ZINC.