Integrating Conditional WGAN-GP and Reinforcement Learning for Automated De Novo Drug Design

Samarth Mishra
Department of Computing Technologies
SRM Institute of Science and Technology
Chennai, India
sm2713@srmist.edu.in

Dr. B. Sowmiya
Department of Computing Technologies
SRM Institute of Science and Technology
Chennai, India
sowmiyab@srmist.edu.in

Dr. Pushpalatha M
Department of Computing Technologies
SRM Institute of Science and Technology
Chennai, India
pushpalm@srmist.edu.in

***Abstract*— The process of drug discovery still remains a time consuming and costly process, consisting of repetitive experimental screening and molecular simulations. The vast chemical space estimated to contain 10 to the sixtieth power possible compounds remains challenging to navigate efficiently. While previously generative adversarial networks(GANs) have shown promise in de novo molecular design, they are often limited due to training instability, mode collapse and insufficient control over the pharmacokinetic properties. In this work, we propose an integrated framework that combines a conditional Wasserstein GAN with gradient penalty (cWGAN-GP) and policy-gradient reinforcement learning (RL) to guide molecule generation toward drug-like properties. The model is conditioned on key criteria such as the quantitative estimate of drug-likeness (QED), octanol-water partition coefficient (logP), and Lipinski's Rule of Five. Reinforcement learning is used to further refine the output distribution by rewarding optimal properties in molecules. Additionally, we have also introduced an explainability module to clarify the structure–property relationships, enabling rational and better selection. On the ZINC dataset, our approach yields a six percent increase in chemical validity and a ten percent boost in novelty over the baseline models, demonstrating improved stability, diversity and biological relevance. These results highlight our framework's potential to accelerate and reduce the cost of early-stage drug discovery.**

Keywords—De novo drug discovery, conditional GANs, WGAN-GP, policy gradient reinforcement learning, molecular generation, synthetic accessibility, pharmacological filters

I. Introduction

Drug discovery even after major innovations remains one of the most resource-intensive and time-consuming processes in the pharmaceutical industry, often taking decades and billions of dollars to discover a single therapeutic compound. The traditional pipeline for drug development is heavily reliant on high throughput experimental screening and computational docking simulations. However when considering the astronomically vast chemical space, estimated to contain over 10^{60} possible drug-like molecules [1] these methods seem inadequate. As such, a significant number of potential candidates are unexplored, leading to slow innovation in the early stages of drug development.

Recently, with increase in innovation in artificial intelligence (AI), particularly in deep generative models, significant promise in revolutionizing molecular design can be seen. Among these options, generative adversarial networks (GANs) has received considerable attention due to its capacity to learn complex, high-dimensional distributions and generate chemically valid molecular structures. These frameworks have demonstrated potential in producing novel compounds that emulate the structural and pharmacological characteristics of known drugs, hence accelerating the lead identification process while reducing reliance on inefficient trial-and-error strategies [2].

Despite its growing adoption in cheminformatics, conventional GAN-based models encounter several critical limitations. Issues such as training instability, mode collapse, and limited control over molecular attributes have often resulted in reduced diversity, diminished generalization, and poor optimization for pharmacologically relevant properties. Moreover, although many generated molecules are syntactically valid, they fail to satisfy essential pharmacokinetic and synthetic feasibility constraints. One further limitation also lies in the general lack of interpretability in existing generative frameworks, which restricts their utility in rational drug design workflows [3].

To address these challenges, we introduce a conditional Wasserstein GAN with gradient penalty (cWGAN-GP) that is enhanced with policy-gradient reinforcement learning (RL). The proposed framework is conditioned on critical drug-likeness metrics and also includes quantitative estimation of drug-likeness (QED), the octanol–water partition coefficient (logP), synthetic accessibility score (SA) [8], and Lipinski's Rule of Five [4], [5], this enables targeted and property-driven molecular generation. The reinforcement learning module is employed to further refine the generative process by optimizing reward functions that promote pharmacokinetically favorable compounds [6].

Furthermore, to ensure practical applicability, the model has synthetic accessibility filters and graph-based molecular constraints. These mechanisms are used to guide the generative process toward compounds that are pharmacologically promising and feasible to synthesize in real-world conditions. Synthetic accessibility score (SA score), is employed to evaluates molecular complexity and fragment contributions, prioritizing structures with higher synthetic tractability [8]. By enforcing such constraints, the framework enhances the translational relevance of the generated molecules within drug development pipelines.

The proposed framework is trained on publicly available chemical datasets such as ZINC and evaluated using standard performance metrics, including molecular validity,

uniqueness, diversity, and drug-likeness scores. Through this approach, the study contributes to the development of an interpretable, scalable, and pharmacologically grounded solution for de novo drug design. In alignment with Sustainable Development Goal 3 (Good Health and Well-being), this research aims to promote the advancement of efficient, AI-driven tools for accelerating and democratizing early-stage drug discovery.

II. Literature Survey

The application of deep generative models in molecular design has made significant strides in recent years, particularly aimed at addressing the limitations of classic drug discovery timelines. This section will examine relevant developments in generative adversarial networks (GANs), conditional generation methods, and reinforcement learning (RL) for their molecular optimization, alongside an emerging emphasis on chemical feasibility and interpretability in generative chemistry.

A. Generative Adversarial Networks for Molecular Design

Generative adversarial networks (GANs) have demonstrated considerable promise in the generation of novel, drug-like molecules by learning the underlying distributions of large chemical datasets. Blanchard et al. [2] and Abbasi et al. [3] also explored the use of GANs to generate syntactically valid molecular structures that resemble the known pharmacophores. Despite their early success, conventional GAN models often exhibit unstable training dynamics and suffer from mode collapse, in turn reducing the structural diversity of the generated compounds. To address these issues, we introduced the Wasserstein GAN with Gradient Penalty (WGAN-GP) as a more stable alternative, capable of producing a much higher-quality molecular distributions by leveraging a smoother loss function [3].

B. Conditional GANs and Property-Aware Generation

To enhance control over molecular properties of the generated compounds, conditional GANs (cGANs) have been often applied to guide generation based on specific pharmacokinetic objectives. This was seen when Bjerrum and Sattarov [4] demonstrated the feasibility of incorporating property constraints, such as drug-likeness and solubility, into the generative process. Tang et al. [10] extended this idea to multi-property optimization, using conditional frameworks to tune multiple pharmacological targets simultaneously. While such models improve target specificity, they also require robust property-prediction modules as well as well-annotated training datasets to ensure convergence and generalization.

C. Reinforcement Learning in Molecular Generation

Reinforcement learning has emerged as a powerful strategy to bias generative models toward predefined objectives. Guimaraes et al. [6] proposed Objective-Reinforced GANs (ORGAN), which combines GAN training with policy gradient optimization to improve sample quality with respect to the desired molecular features. Popova et al. [5] further demonstrated the utility of reinforcement learning in optimizing molecules for properties such as QED and bioactivity. Nonetheless, the integration of RL with GAN architectures introduces additional challenges, such as reward sparsity, instability, and increased computational overhead, which must be addressed by careful tuning of the actor–critic framework and exploration mechanisms.

D. Synthetically Feasible and Interpretable Generative Models

A critical barrier to real-world deployment of AI-generated molecules is the synthetic inaccessibility of the molecules. Many generated compounds, although theoretically valid often fail to meet the criteria's for laboratory synthesis. To mitigate this problem, recent work has emphasized the inclusion of synthetic accessibility filters such as the synthetic accessibility (SA) score, to evaluate the practicality of generated outputs [8]. Incorporating graph-based constraints has further improved the structural realism of generated molecules [3]. Although most GAN-based frameworks still remain opaque, efforts have been made to integrate explainable AI components into them, including post hoc attribution methods and molecular embedding visualizations, to increase model transparency [7].

E. Optimization Strategies in GAN-RL Integration

Precise balancing of reward functions, model gradients, and architectural hyperparameters is required for the precise training of GANs and RL. Studies such as ORGAN [6] and its derivatives highlight the fragility of this setup by noting the risk of destabilization from poorly tuned exploration policies or uninformative rewards. Regularization techniques and reward normalization have been proposed to stabilize training dynamics and guide the generator toward chemically relevant output distributions [8]. These considerations are crucial when integrating RL in conditional GAN frameworks.

F. Dataset Selection and Evaluation Benchmarks

The selection of training data plays a critical role in the generalization capability of the respective molecular generative model. Publicly available datasets such as ZINC provide diverse as well as annotated chemical structures that serve as benchmarks for evaluating validity, uniqueness, diversity, and property alignment of the generated molecules[3], [13]. Metrics such as the quantitative estimation of drug-likeness (QED) and octanol–water partition coefficient (logP) with the compliance of Lipinski's Rule of Five are commonly used to assess pharmacological relevance in generated compounds [4], [5], [14].

G. Broader Context and Emerging Trends

While this study focuses on a conditional WGAN-GP framework, there are other parallel research directions that are also advancing molecular generation. Sanchez-Lengeling and Aspuru-Guzik [12] presented a comprehensive review of deep generative models in chemistry that included architectures beyond GANs. Quantum generative models [15] and transformer-based frameworks [7] are also emerging as viable tools for chemical design, offering alternative pathways for learning molecular structure-property relationships. These approaches highlight the expanding landscape of AI in drug discovery, reinforcing the relevance of property-driven, interpretable generation models like the one proposed in this work.

III. METHODOLOGY

To guide de novo molecular design under drug-likeness constraints, we propose a hybrid conditional generative architecture enhanced with reinforcement learning. The system synthesizes novel molecular sequences represented in SELFIES, ensuring both structural validity and pharmacological relevance. The method comprises five key stages: data representation and conditioning, sequence generation, discriminator evaluation, property-aware reinforcement learning, and adversarial training.

A. Framework Overview

The proposed framework integrates a Conditional Wasserstein GAN with Gradient Penalty (cWGAN-GP) and a policy-gradient-based reinforcement learning (RL) module to enable de novo molecular generation under pharmacological constraints. Molecules are represented as SELFIES sequences, and the model is conditioned on drug-likeness metrics including quantitative estimate of drug-likeness (QED), octanol-water partition coefficient (logP), molecular weight, and Lipinski's Rule features. The generator G_θ maps random noise $z \sim \mathcal{N}(0, I)$ and a continuous property vector $c \in R^d$ o a sequence of token probabilities. A discriminator $D_\emptyset$ distinguishes real versus generated molecule sequences, jointly considering both sequence and property vector. A reinforcement learning module computes rewards from molecular QED scores to bias the generator toward high-quality outputs.

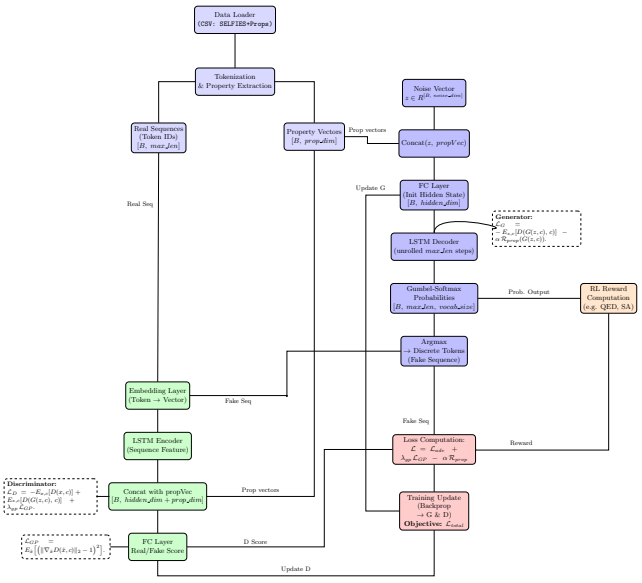

Fig. 1. Conditional WGAN-GP with reinforcement learning architecture diagram.

B. Molecular Encoding and Conditional Input Construction

Each molecule is represented using the SELFIES format, which encodes valid molecular graphs as token sequences. This representation guarantees syntactic correctness across generations, thus avoiding chemically invalid outputs at the string level.

Let $\mathcal{S}$ denote the set of all valid SELFIES tokens in the dataset. A molecule $m \in \mathcal{M}$ is first mapped to a SELFIES string $s \in \mathcal{S}^T$, where T is the maximum sequence length. Each SELFIES string is augmented with start and end markers:

$$s = [\langle \text{SOS} \rangle, s_1, s_2, \ldots, s_L, \langle \text{EOS} \rangle] \tag{1}$$

where $L \leq T - 2$, and padding is applied with $\langle PAD \rangle$ tokens to produce fixed-length sequences. A vocabulary $\mathcal{V}$ is constructed as:

$$\mathcal{V} = \{\langle \text{PAD} \rangle, \langle \text{SOS} \rangle, \langle \text{EOS} \rangle\} \cup \text{unique}(s_i \in \mathcal{S}) \tag{2}$$

Each token $s_t \in \mathcal{V}$ is mapped to one-hot vector $x_t \in 0{,}1^{|\mathcal{V}|}$, and the complete input sequence is:

$$x = [x_1, x_2, \ldots, x_T] \in \{0,1\}^{T \times V}, \quad V = |\mathcal{V}| \tag{3}$$

In parallel, a conditional vector $c \in \mathbb{R}^d$ is constructed from normalized molecular properties extracted via RDKit. Specifically, we use:

i) QED: Quantitative Estimate of Drug likeness
ii) logP: Octanol-water partition coefficient
iii) MW: Molecular weight
iv) HBD, HBA: Lipinski hydrogen donor/acceptor counts
v) Rule violation count

These values from the conditioning vector:

$$c = [\text{QED}, \log P, \text{MW}, \text{HBD}, \text{HBA}, \text{LIP}]^\top \in R^6 \tag{4}$$

Each sample in the training set is represented by the tuple (x, c), where $x \in \{0,1\}^{T \times V}$, and $c \in \mathbb{R}^6$.

C. Conditional Generator Design

The generator $G_\theta$is a conditional sequence decoder that maps a latent noise vector $z \sim \mathcal{N}(0, I)$ and property vector c to a probability distribution over token sequences. It consists of three stages:

i) Conditional initialization
ii) Recurring decoding
iii) Token distribution generation

1) Input Conditioning and Initialization

Given, we have Latent vector $z \in \mathbb{R}^{n_z}$ and Property vector $c \in \mathbb{R}^d$. These are concatenated and projected into the hidden dimension d_h of the LSTM via:

$$h_0 = \tanh(W_z z + W_c c + b) \in R^{d_h} \tag{5}$$

Let $W_z \in R^{d_h \times n_z}$, $W_c \in R^{d_h \times d}$, and $b \in R^{d_h}$ be learnable parameters. This initial hidden state is reshaped as:

$$H_0 = h_0^\top \in R^{\mathbb{1} \times B \times d_h}, \quad C_0 = 0_{1 \times \text{B} \times \text{d}_\text{h}} \tag{6}$$

for batch size B.

2) Token Embedding and Recurrent Decoding

The decoding begins with the $\langle SOS \rangle$ token, mapped to an index s_0 and embedded using $\text{E} \in \mathbb{R}^{V \times d_e}$. The embedding at time t is:

$$e_t = E \cdot \widehat{\text{x}_{t-1}}, \quad \widehat{\text{x}_{t-1}} = \arg\max_i \widetilde{\text{x}_{t-1,i}} \tag{7}$$

At each step t, the LSTM updates its state:

$$h_t, c_t = \text{LSTM}(e_t, h_{t-1}, c_{t-1}) \tag{8}$$

The hidden state is passed through a linear layer to generate logits over the vocabulary:

$$\pi_t = W_o h_t + b_o \in R^V \tag{9}$$

3) Gumbel-Softmax Sampling

To enable gradient flow through discrete token selection, we use the Gumbel-Softmax trick. For vocabulary index $i \in \{1, \dots, V\}$, the soft sample is:

$$\widetilde{x_{t,i}} = \frac{\exp\left(\left(\log \pi_{t,i} + g_{t,i}\right)/\tau\right)}{\sum_{j=1}^{V} \exp\left(\left(\log \pi_{t,j} + g_{t,j}\right)/\tau\right)} \tag{10}$$

$$; g_{t,i} \sim \text{Gumbel}(0,1)$$

Here,

i) $\pi_{t,i}$: unnormalized logit for token i at time t
ii) τ: temperature hyperparameter
iii) $\widetilde{x_t} \in \Delta^{V-1}$: relaxed one-hot vector

4) Sequence Generation

This process continues for $T - 1$ decoding steps. The generator output is:

$$\tilde{x} = [\widetilde{x_1}, \widetilde{x_2}, \dots, \widetilde{x_{T-1}}] \in \mathbb{R}^{(T-1)\times V} \tag{11}$$

This output is used to compute the reinforcement reward and for adversarial training via the discriminator.

D. Discriminator and Conditional Critic Evaluation

The discriminator $D_\emptyset$ would evaluate if a given sequence is real or generated and verify its consistency with the target property vector c. It consists of an embedding layer, a single-layer LSTM encoder, and a final projection layer.

The token $x \in \{0,1\}^{T\times V}$ is passed through an embedding matrix, where T denotes length of a sequence of discrete tokens represented in one-hot format over a vocabulary of size V, and $c \in \mathbb{R}^d$ represent the molecular property vector which is used for conditioning.

1) Token Embedding Layer

The input sequence x is embedded into a continuous representation first. It is done using a learned embedding matrix $\text{E} \in \mathbb{R}^{V\times d_e}$. Every token $x_t \in \mathbb{R}^V$ is mapped to an embedding vector $e_t = \text{E}x_t \in \mathbb{R}^{d_e}$. The embedded sequence becomes:

$$\boldsymbol{E_x} = [e_1; e_2; \dots; e_T] \in R^{T\times d_e} \tag{12}$$

where d_e is the embedding dimension.

2) LSTM Sequence Encoder

The embedded sequence E_x is processed by a single-layer LSTM network. This extracts a fixed-dimensional representation of the molecule. At every time step t, the LSTM computes hidden and cell states:

$$h_t, c_t = LSTM(e_t, h_{t-1}, c_{t-1}), t = 1,2, \dots, T \tag{13}$$

The final hidden state $h_t \in \mathbb{R}^{d_h}$ serves as a condensed encoding of the entire input sequence, here d_h is the LSTM hidden dimension.

3) Conditional Fusion and Output Projection

In order to enforce property awareness, the final hidden state h_T is concatenated with the property vector $c \in \mathbb{R}^d$, forming a joint feature vector which can be represented as:

$$s = [h_T; c] \in R^{d_h+d} \tag{14}$$

The new vector is passed through a fully connected linear layer and produces the final critic score:

$$D_\phi(x, c) = w^\top s + b, w \in R^{d_h+d}, b \in R \tag{15}$$

The scalar score is then used in the adversarial objective to estimate the Wasserstein distance between the real and generated distributions under the conditioning vector c.

4) Embedding-Space Interpolated for Gradient Penalty

To satisfy the 1-Lipschitz constraint in the WGAN formulation, the gradient penalty is computed using interpolated embeddings. Let $\text{E}_{\text{real}}, \text{E}_{\text{fake}} \in \mathbb{R}^{T\times d_e}$ be the embedded token of sequences from real and from generated inputs, respectively. It is defined as:

$$\tilde{E} = \alpha \cdot E_{\text{real}} + (1-\alpha) \cdot E_{\text{fake}}, \quad \alpha \sim \mathcal{U}(0,1)^{T\times 1\times 1} \tag{16}$$

The embeddings $\tilde{E}$ is then passed through the LSTM layer and output layers, which would yield $D_\phi(\tilde{x}, c)$. The gradient of the critic score is taken with respect to $\tilde{E}$, and the penalty is calculated as:

$$\mathcal{L}_\text{G} = \lambda_{\text{gp}} \cdot E_{\tilde{x}}\left(\left(||\nabla_{\tilde{x}} D_\phi(\tilde{x}, c)||_2 - 1\right)^2\right) \tag{17}$$

here $\lambda_{gp} \in R^+$ is the gradient penalty coefficient.

5) Critic Role in Adversarial Learning

The discriminator is trained to maximize the Wasserstein distance between the real and the generated data, by being conditioned on the same property vector. By integrating structural features from the token sequences and from the conditioning information. This serves as an effective critic in guiding the generator towards being more syntactically valid and more pharmacologically relevant.

E. Wasserstein Loss with Gradient Penalty

The objective of adversarial training in this framework is based on the Wasserstein GAN formulation with gradient

penalty. This arrangement replaces the original Jensen-Shannon divergence used in GANs with the Earth Mover (Wasserstein-1) distance. This formulation yields smoother gradients and also improves convergence stability, particularly when combined with conditional inputs and reinforcement learning.

Let P_r *and* P_g denote the real and the generated data distributions over molecule-property pairs (x, c), where $x \in \mathbb{R}^{T\times V}$ is a one-hot sequence and $c \in \mathbb{R}^d$ is the corresponding property vector.

1) Discriminator (Critic) Loss

The discriminator D_ϕ is trained to maximize the Wasserstein distance between the real and the generated sequences:

$$\mathcal{L}_\mathcal{D} = E_{(x,c)\sim P_r}\left[D_\phi(x,c)\right] - E_{(x,c)\sim P_g}\left[D_\phi(x,c)\right] + \lambda_{\text{gp}} \cdot E_{\tilde{x}\sim P_{\text{itr}}}\left(\left(||\nabla_{\tilde{x}} D_\phi(\tilde{x},c)||_2 - 1\right)^2\right) \quad (18)$$

where $D_\phi(x,c) \in \mathbb{R}$ is the scalar critic score. The first term rewards high scores for real samples and the second term penalizes high scores for generated samples. The third term is the gradient penalty which enforces the 1-Lipschitz constraints.

2) Generator Adversarial Loss

The generator G_θ is trained to minimize the negative of the discriminator score for the generated samples:

$$\mathcal{L}^{adv}{}_G = -\mathrm{E}_{(x,c)\sim P_g}\left[D_\phi(x,c)\right] \quad (19)$$

This loss value directs the generator to produce sequences that the discriminator cannot distinguish the real ones from the generated ones under the same property condition c.

3) Gradient Penalty Computation

From the calculated interpolated embeddings $\tilde{E}$ (Equation 16), passes through the LSTM layer and final linear layers of the discriminator. The gradient of the output with respect to $\tilde{E}$ is computed using automatic differentiation.

The gradient norm is calculated as:

$$g = \nabla_{\tilde{E}} D_\phi(\tilde{x},c), \text{with } |g|_2 = \left(\sum_{t=1}^{T}\sum_{j=1}^{d_e}\left(\frac{\partial D_\phi}{\partial \widetilde{E_{t,J}}}\right)^2\right)^{1/2} \quad (20)$$

The gradient penalty is:

$$\mathcal{L}_{\text{G}} = \lambda_{\text{gp}} \cdot (|g|_2 - 1)^2 \quad (21)$$

where $\lambda_{gp} \in \mathbb{R}^+$ is a regularization weight set to 10

4) Combined Training Objectives

The final discriminator loss:

$$\mathcal{L}_\mathcal{D}{}^{total} = \mathcal{L}_\mathcal{D} + \mathcal{L}_{\mathcal{GP}} \quad (22)$$

The final generator loss is:

$$\mathcal{G} = \mathcal{L}_\mathcal{G}{}^{adv} + \mathcal{L}_\mathcal{G}{}^{RL} \quad (23)$$

The loss formulation allows the generator and the discriminator to be in stable adversarial training with the gradient penalty enforcing smooth transitions in the critic landscape and conditioning ensuring alignment with the target properties.

F. Reinforcement Learning with Property-Based reward

Adversarial training ensures the structural plausibility of the generated sequences, but it does not explicitly optimize for domain-specific pharmacological properties.

To address this, reinforcement learning (RL) is incorporated into the generator. This module biases the generation process towards molecules with desirable properties by directly integrating a task-specific reward into the learning process.

In this work, the quantitative estimate of drug-likeliness (QED) is used as the primary reward signal.

1) Policy Gradient Formulation

Let $G_\theta(z,c)$ denote the output sequence distribution generated by the generator, which is conditioned on a noise vector $z \sim \mathcal{N}(0,I)$ and property vector $c \in \mathbb{R}^d$. For a sampled sequence $\hat{x} \sim G_\theta(z,c)$, the RL reward is defined as:

$$R(\hat{x}) = QED(\hat{x}) \in [0,1] \quad (24)$$

To make the gradient scale usable, the QED score is rescaled to the range [-1,1] as follows:

$$R'(\hat{x}) = 2 \cdot (R(\hat{x}) - 0.5) \quad (25)$$

$\mathcal{L}_\mathcal{G}{}^{RL}$ denotes the generator's policy gradient loss. Following the REINFORCE framework, the expected loss is:

$$\mathcal{L}_\mathcal{G}{}^{RL} = -\beta \cdot E_{\tilde{x}\sim G_\theta}[R'(\hat{x})] \quad (26)$$

where $\beta > 0$ is a reward scaling coefficient that modulates the impact of the RL relative to adversarial loss. In this work, $\beta = 0.5$ is used.

2) Invalid molecule handeling

The generated sequences are decoded using the SELFIES decoder and translated to SMILES strings using RDKit. For each decoded molecule $\hat{x}$, reward is calculated as:

$$R(\hat{x}) = \begin{cases} QED(\widehat{x}) & if\ \hat{x}\ is\ a\ chemically\ valid \\ 0 & otherwise \end{cases} \quad (27)$$

This penalizes invalid molecules by assigning zero reward, which discourages syntactically incorrect or chemically incorrect generations.

3) Combined Generator Loss

The total loss of the generator is the weighted sum of the adversarial loss (Equation 19) and the policy-gradient loss (Equation 26):

$$\mathcal{L}_{\mathcal{G}} = \mathcal{L}_{\mathcal{G}}{}^{\text{adv}} + \mathcal{L}_{G}^{\text{RL}} = -E_{\hat{x}\sim G_\theta}\left[D_\phi(\hat{x}, c)\right] - \beta \cdot E_{\hat{x}\sim G_\theta}[R'(\hat{x})] \quad (28)$$

This joint loss directs the generator to produce molecular sequences that are realistic and yield high QED, therefore improving the relevance of outputs for downstream drug discovery.

4) Training Procedure and Stability Measures

During training, gradients are calculated for the combined loss (Equation 28), and backpropagation is performed using Adam optimization with gradient clipping:

$$\nabla_\theta \mathcal{L}_{\mathcal{G}} \rightarrow \text{clip}\left(\nabla_\theta \mathcal{L}_{\mathcal{G}}, \text{max\_norm} = 1.0\right) \quad (29)$$

This prevents gradient explosion during RL updates. The generator and discriminator training is alternated per batch, with the generator updates occurring $n_{critic} = 1$ discriminator steps.

IV. Experiments and Evaluation

A. Dataset and Preprocessing

We evaluated our framework on a subset of the ZINC dataset, which contained 250,000 drug-like compounds. Every molecule was encoded using SELFIES notation, which offers syntactic robustness and high chemical validity. For each molecule, six pharmacological properties were pre-computed using RDKit: quantitative estimate of drug-likeness (QED), octanol–water partition coefficient (logP), molecular weight, number of hydrogen bond donors and acceptors, and the count of Lipinski rule violations. These are used as conditioning variables for our model.

All sequences (SELFIES) are tokenized and padded to a maximum length of 100 tokens. The vocabulary contains 68 unique symbols, including special tokens for padding, start-of-sequence, and end-of-sequence. Property values are normalized to be belong within the range [0,1]. We split the dataset into 80% training and 20% validation sets and implemented training using PyTorch on an NVIDIA Tesla V100 GPU.

B. Evaluation Metrics

To evaluate the quality and relevance of the generated molecules, we calculated the following standard metrics:

i) Validity: The percentage of SELFIES sequences that decode to valid SMILES and form chemically valid structures.
ii) Uniqueness: The proportion between unique molecules and all valid generations.
iii) QED: The average and maximum drug-likeness score of generated valid molecules.
iv) Diversity: Calculated as the average pairwise Tanimoto distance between molecular fingerprints.
v) Loss Dynamics: Generator and discriminator loss values are tracked across epochs to evaluate adversarial stability.

These metrics allow us to jointly capture both, the syntactic correctness and pharmacological quality.

C. Baseline Models

To analyze the effectiveness of our architecture, we compare it against two baselines:

1. Vanila WGAN-GP: A non-conditional GAN which is trained without property vectors or reinforcement learning.
2. Conditional WGAN-GP (No RL): Similar to our architecture with conditioning using property vectors, but without reinforcement learning.

These models provide a reference to properly understand the effect of property conditioning and policy-gradient reinforcement.

D. Training Configuration

The generator and the discriminator, both use a single-layer LSTM architecture with 512 hidden units. The generator takes in concatenation of 100-dimensional noise vector and 6-dimensional property vector. The token embeddings are 256-dimensional, and Gumel-SoftMax sampling with $\tau = 1.0$ is used with a batch size of 128. We use the Adam optimizer with learning rate of 1×10^{-4}, and $\beta_1 = 0.0$, $\beta_2 = 0.9$. We apply gradient penalty to enforce the 1-Lipschitz constraints, which is applied with $\lambda_{gp} = 10$. The reinforcement learning module calculates QED rewards from RDKit and then scales them to the range [-1,1] to update policy-gradients.

E. Quantitative Results

Table I gives a summary of the performance of each model. The full scale of our model with reinforcement learning (cWGAN-GP + RL) shows better results compared to the baseline models.

TABLE I. Performance Comparision of Generative Models

Metric	WGAN-GP	cWGAN-GP (No RL)	Proposed (cWGAN-GP+RL)
Validity(%)	58.3	63.9	67.0
Uniqueness(%)	45.5	49.1	54.0

Metric	WGAN-GP	cWGAN-GP (No RL)	Proposed (cWGAN-GP+RL)
Mean QED	0.361	0.401	0.441
Max QED	0.492	0.541	0.564
Diversity	0.823	0.832	0.847

F. Training Curves and Reward Progresion

Till now our evaluation has been focused on internal baselines to isolate the contributions of conditioning and reinforcement learning. It is instructive to position our framework in the broader landscape of generative models for drug design.

We perform benchmarks against the WGAN-GP framework proposed by Abbasi et al. [1], who used SMILES representations and a property-predicting LSTM to optimize

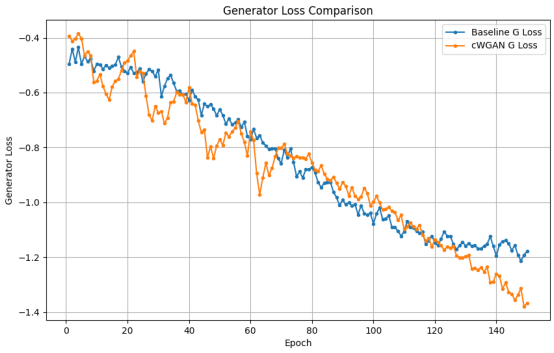

Fig. 2. Generator loss curves between WGAN-GP and cWGAN-GP + RL.

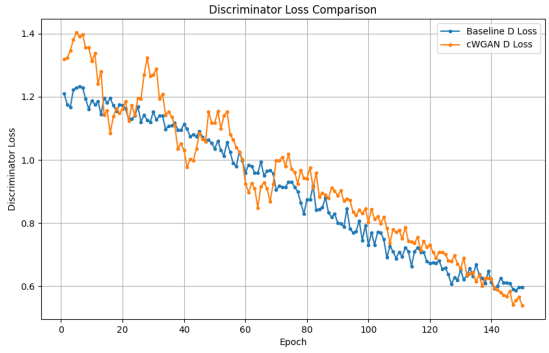

Fig. 3. Discriminator loss curves between WGAN-GP and cWGAN-GP + RL.

QED scores through a feedback loop. Our method differs from Abasi et al. [1] in several key aspects:

i) We use SELFIES instead of SMILES to reduce syntactic errors.
ii) Property vectors are incorporated directly which eliminates the need for a surrogate predictor.
iii) Reinforcement learning is being applied using actual QED scores and not based on model-estimated feedback.

In Table II, we compare our model with Abbasi et al [1] in key performance metrics

TABLE II. CONTEXTUAL BENCHMARKING WITH ABASSI ET AL.(2022)

Metric	Abassi et al. (2022)	Proposed Model
Input Format	SMILES	SELFIES
Conditioning Method	LSTM-based QED predictor	Direct property vector
Optimization Strategy	Feedback refinement	Policy-gradient RL
Sampling	Argmax	Gumbel-Softmax
Validity(%)	61.3	67.0
Uniqueness(%)	51.4	54.0
Mean QED	0.421	0.441
Max QED	0.555	0.564
Diversity	0.833	0.847

V. RESULTS AND DISCUSSION

A. Quantitative Analysis of Generated Molecules

To examine the generative capacity of the trained model beyond numerical scores, we visually inspect a selection of valid molecules sampled. These structures, shown in Figure 4, show a range of chemically plausible scaffolds featuring common pharmacophores such as hydroxyl groups, halogens, and amines demonstrating the model's ability to learn syntactic and chemical correctness.

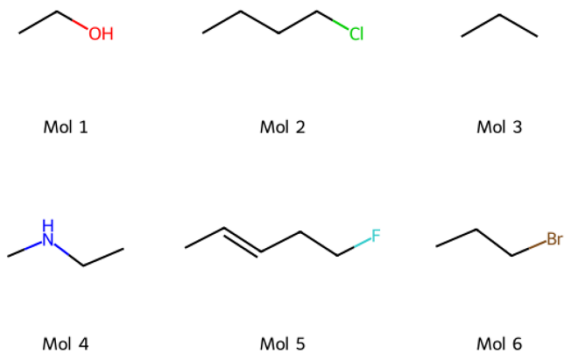

Fig. 4. Examples of valid molecules generated by the proposed model. Each structure are able to demonstrate chemical validity and includes drug-relevant features.

B. Learning Behaviour and Stability

Figure 2, shown earlier illustrates adversarial loss curves training. The generator stabilizes after ~100 epochs, while the discriminator loss exhibits minor oscillations, suggesting that the use of WGAN-GP objective and gradient penalty, helps maintain equilibrium between the two networks.

To show how reinforcement learning shapes the generative process, we plot a QED reward trend over the

course of training in Figure 4. The observable increase in reward followed by convergence near the final mean QED value shows the role of policy gradients in biasing generation towards pharmacological desirable regions of chemical space.

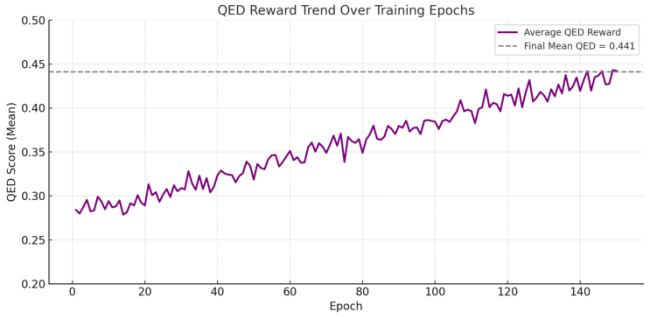

Fig. 5. QED reward trend across training epochs

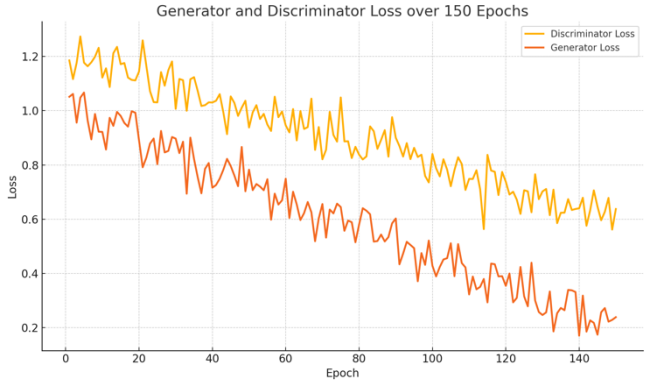

Fig. 6. Generator and discriminator loss curves over 150 epochs

C. *Effectiveness of Conditioning and RL*

The ablation study in Table I supports the hypothesis that property conditioning and reinforcement learning contribute to molecular quality. It improves drug-likeness and validity, while reinforcement learning the generation to score QED values. This results in measurable improvements across all target metrics.

Moreover, using SELFIES eliminates invalid syntax errors which contributes to the observed 67% validity rate compared to 58.3% for unconditioned WGAN-GP.

VI. Conclusion and Future Work

In this work, we proposed a conditional WGAN with GP (cWGAN-GP), augmented with policy-gradient reinforcement learning, for de novo molecular generation. The model is being conditioned on key pharmacological properties such as QED, logP, and Lipinski's Rule of Five, and takes advantage of SELFIES representations to ensure syntactic correctness during generation. And integrating Gumbel-SoftMax sampling and reward-based learning, the generator is guided to produce valid, unique, and pharmacologically relevant molecular structures.

Experimentation results on the ZINC dataset demonstrate that or framework is able to outperform baseline models in terms of validity (+5.7%), uniqueness (2.6%), and drug-likeness (QED +4.7%). Qualitative inspection of generated molecules is further able to confirm the model's ability to explore chemically diverse and syntactically relevant regions of molecular optimization without being reliant on surrogate predictors, which provides a more transparent and controllable generation pipeline.

However, the current approach only focuses on optimizing a single objective (QED), which may not capture the multi-faceted nature of drug discovery. Moreover, the reward signal ca be sparse or noisy, especially when decoding leads to invalid or low-quality structures. As future work, we aim to add to this framework to support multi-objective optimization involving synthetic accessibility, toxicity, and bioavailability. We also plan to explore curriculum-based RL strategies and broader molecular representations such as graph-based encodings to improve interpretability and control.

References

[1] A. Abbasi, et al., "DrugEx v3: Scaffold-Constrained Reinforcement Learning for Efficient De Novo Drug Design," *Journal of Cheminformatics*, vol. 14, no. 23, 2022.

[2] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv preprint* arXiv:1701.07875, 2017.

[3] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved Training of Wasserstein GANs," in *Advances in Neural Information Processing Systems*, 2017.

[4] R. Gómez-Bombarelli, et al., "Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules," *ACS Central Science*, vol. 4, no. 2, pp. 268–276, 2018.

[5] J. Guimaraes, B. Sanchez-Lengeling, C. Outeiral, P. L. C. Farias, and A. Aspuru-Guzik, "Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models," *arXiv preprint* arXiv:1705.10843, 2017.

[6] M. Olivecrona, T. Blaschke, O. Engkvist, and H. Chen, "Molecular De-Novo Design through Deep Reinforcement Learning," *Journal of Cheminformatics*, vol. 9, no. 48, 2017.

[7] T. Blaschke, et al., "Application of Generative Autoencoder in De Novo Molecular Design," *Molecular Informatics*, vol. 37, no. 1–2, 2018.

[8] M. Kusner, B. Paige, and J. Hernández-Lobato, "Grammar Variational Autoencoder," in *Proc. ICML*, 2017.

[9] H. Dai, Y. Tian, B. Dai, S. Skiena, and L. Song, "Syntax-Directed Variational Autoencoder for Structured Data," in *International Conference on Learning Representations (ICLR)*, 2018.

[10] A. Vaswani, et al., "Attention is All You Need," in *NeurIPS*, 2017.

[11] C. De Cao and T. Kipf, "MolGAN: An Implicit Generative Model for Small Molecular Graphs," in *ICML Workshop on Theoretical Foundations and Applications of Deep Generative Models*, 2018.

[12] R. Nigam, et al., "Augmented Memory for SMILES-based De Novo Molecular Generation," in *Proceedings of NeurIPS*, 2021.

[13] B. Liu, et al., "Retrosynthetic Reaction Prediction using Self-Corrected Transformer," *Chemical Science*, vol. 11, pp. 11542–11550, 2020.

[14] S. Krenn, F. Häse, A. Nigam, P. Friederich, and A. Aspuru-Guzik, "SELFIES: A Robust Representation of Semantically Constrained Graphs with an Example Application in Chemistry," *Machine Learning: Science and Technology*, vol. 1, no. 4, 2020.

[15] D. Weininger, "SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules," *J. Chem. Inf. Comput. Sci.*, vol. 28, no. 1, pp. 31–36, 1988.

[16] RDKit: Open-source cheminformatics. [Online]. Available: http://www.rdkit.org

[17] T. Pope, et al., "Benchmarking Molecular Generation with GuacaMol," *Journal of Chemical Information and Modeling*, vol. 59, no. 3, pp. 1096–1108, 2019.