molecules

MDPI

Article

Generation of Rational Drug-like Molecular Structures Through a Multiple-Objective Reinforcement Learning Framework

Xiangying Zhang, Haotian Gao, Yifei Qi, Yan Li * and Renxiao Wang *

Department of Medicinal Chemistry, School of Pharmacy, Fudan University, 826 Zhangheng Road, Shanghai 201203, China

* Correspondence: li_yan@fudan.edu.cn (Y.L.); wangrx@fudan.edu.cn (R.W.)

Abstract: As an appealing approach for discovering novel leads, the key advantage of de novo drug design lies in its ability to explore a much broader dimension of chemical space, without being confined to the knowledge of existing compounds. So far, many generative models have been described in the literature, which have completely redefined the concept of de novo drug design. However, many of them lack practical value for real-world drug discovery. In this work, we have developed a graph-based generative model within a reinforcement learning framework, namely, METEOR (Molecular Exploration Through multiplE-Objective Reinforcement). The backend agent of METEOR is based on the well-established GCPN model. To ensure the overall quality of the generated molecular graphs, we implemented a set of rules to identify and exclude undesired substructures. Importantly, METEOR is designed to conduct multi-objective optimization, i.e., simultaneously optimizing binding affinity, drug-likeness, and synthetic accessibility of the generated molecules under the guidance of a special reward function. We demonstrate in a specific test case that without prior knowledge of true binders to the chosen target protein, METEOR generated molecules with superior properties compared to those in the ZINC 250k data set. In conclusion, we have demonstrated the potential of METEOR as a practical tool for generating rational drug-like molecules in the early phase of drug discovery.

Keywords: molecular generative model; de novo drug design; multi-objective optimization; GCPN

check for updates

Academic Editors: Shijun Zhong and Manuel A. Coimbra

Received: 14 November 2024
Revised: 4 December 2024
Accepted: 15 December 2024
Published: 24 December 2024

Citation: Zhang, X.; Gao, H.; Qi, Y.; Li, Y.; Wang, R. Generation of Rational Drug-like Molecular Structures Through a Multiple-Objective Reinforcement Learning Framework. *Molecules* **2025**, *30*, 18. https://doi.org/10.3390/molecules30010018

1. Introduction

Virtual screening of compound libraries has been a widely adopted approach in structure-based drug discovery for finding novel lead compounds. However, the potential exploration of molecules with desired properties is severely curtailed by the limited size of available compound libraries (~10^9) [1]. This constraint pales in comparison with the vast chemical space of "drug-like" compounds, which is estimated to range from 10^{23} to 10^{60} [2]. To bridge this gap, de novo drug design offers another approach to delving into the chemical space beyond existing compounds. Conventional de novo design methods typically rely on a pre-defined fragment library to construct molecular structures in a stepwise manner. Such a building-up process is relatively time-consuming, and yet the structural diversity among the generated molecular structures is in principle limited by the fragment library employed therein. Moreover, conventional de novo design methods often produce molecular structures that are challenging to synthesize due to extensive enumeration [3,4]. All these obstacles have hindered the wide application of de novo design to practical drug discovery efforts.

In recent years, generative models, a type of unsupervised training model, have emerged as invaluable tools in various scientific domains [5]. Such models have been able to generate new samples by comprehending the essential probability distribution underlying the given training samples. Generative models quickly found their applications in the realm of chemistry, where they were typically trained on large compound libraries to capture the intrinsic probability distribution embedded in the molecular structures. By drawing samples from the learned distribution, novel molecular structures were generated, which effectively expanded the accessible chemical space. It has been demonstrated that even a tiny fraction, for example, 0.1%, of a compound library, when used to train a generative model, could cover a significant portion of the chemical space spanned by the entire library [6]. Thus, generative models hold great promise in expanding the arsenal for drug discovery. Particularly for de novo drug design, generative models can not only create new molecules but also to craft molecules of specific interest.

Reinforcement learning presents an approach for achieving targeted molecule generation [7]. Within the framework of reinforcement learning, an agent engages with an environment through a sequence of actions. The agent iteratively refines its policy to maximize cumulative rewards across the action sequence, guided by the environment's feedback. In the context of de novo drug design employing reinforcement learning, an environment is tailored to provide rewards to the agent based on the properties of the generated molecules. Previous studies have demonstrated the utility of reinforcement learning in biasing generative models toward the creation of molecules with desired optimized properties [8–18].

However, many of the current generative models exhibit certain limitations when being evaluated in real-world drug discovery scenarios. For example, some models aim at overly contrived objectives, such as maximization of log *P* without any limit [8,9,12,17,19]. Some other models focus exclusively on the binding affinity against a specific target [10,11,14–16]. However, a successful drug discovery process is multi-objective in nature, where one has to consider and evaluate multiple properties of the candidates simultaneously [20–23]. Therefore, we believe that a generative model with practical value for de novo drug design has to be trained in a multi-objective manner.

Accordingly, we have developed such a molecular generative model, namely, METEOR (Molecular Exploration Through multiplE-Objective Reinforcement). METEOR is integrated with a reinforcement learning framework, which allows the rapid design of molecules with desirable drug-likeness and synthetic accessibility, as well as binding affinity to a user-defined target protein. In METEOR, we employ the Graph Convolutional Policy Network (GCPN) originally proposed by You et al. [8] as the fundamental architecture to construct the backend generative model. We evaluated several graph traversal algorithms [24] in terms of their efficiency in molecular structure generation. We also introduced chemical rules to detect improper substructures, thereby substantially elevating the quality of the molecular structures generated. Importantly, we introduced a special reward function to promote multi-objective optimization, combining considerations of binding affinity to the target protein, drug-likeness, and synthetic accessibility. Here, binding affinity to the target protein was evaluated by PLANET, a GNN-based deep learning model developed by our group [25]. Finally, we showcased the potential application of METEOR to real-world drug discovery with a retrospective example.

2. Results and Discussion

2.1. Comparison of Generative Models Based on Different Algorithms

The several generative models developed in our study were trained on the ZINC 250k data set in order to generate valid molecular graphs. To evaluate the performance of these models in this aspect, metrics encompassing validity, uniqueness, and novelty were considered. These metrics were assessed based on a sample of 50,000 molecules generated by each model.

The validity of the generated molecules remained at 100% across all models (Table 1), which should be attributed to the step-by-step valency check enabled during graph generation. In contrast to SMILES-based models, which might encounter validity issues due to syntax problems, graph-based generative models benefit from a more natural representation of molecular structures, ensuring high validity. However, a validity check by RDKit does not guarantee drug-like molecular structures (Figure S1 in the Supporting Information). Thus, we have implemented additional chemical rules in the molecular generation environment in our model to filter out undesired substructures, including cumulative alkenes and peroxyl bonds, double or triple bonds in three or four-membered rings, bridged rings formed with aromatic rings, and large rings. Our analysis indicated that these additional substructure detections led to the elimination of approximately 40% of the impractical structures generated by $\text{GCPN}_{\text{origin}}$. Moreover, the breadth-first model (BFM) was observed to have a problem with ring closure (Figure S2 in the Supporting Information). This problem arose due to the divergent nature of breadth-first graph generation, where the generative model tended to generate molecular graphs with incomplete rings, which may be closed later after several inconsecutive actions. In contrast, GCPN and the depth-first model (DFM) in principle can generate molecular graphs with rings in a more practical and complete manner.

Table 1. Metrics of 50,000 molecules generated by several generative models.

Models	Validity			Uniqueness		Novelty	
	RDKit	Pattern	Completeness	Molecule	Scaffold	Molecule	Scaffold
GCPN (origin)	1.000	0.592	0.993	1.000 [a]	0.666 [a]	1.000 [a]	0.928 [a]
GCPN (ours)	1.000	1.000	0.960	1.000	0.737	1.000	0.953
DFM	1.000	1.000	0.987	0.912	0.626	0.999	0.914
BFM	1.000	1.000	0.677	0.776	0.454	1.000	0.925

[a]: Molecules containing improper substructures are viewed as invalid.

Among our evaluation metrics, uniqueness reflects the fraction of non-duplicate molecules, while novelty reflects the fraction of generated molecules not presented in the training set. Our results show that BFM exhibits the lowest performance in terms of uniqueness and novelty (Table 1). By analyzing BFM-generated molecules, we have observed that the graph generation process is prone to terminate prematurely and produce simple and duplicate structures. In order to evaluate the scaffold uniqueness and novelty presented by the molecules in the ZINC 250k data set, we extracted the Bemis–Murcko scaffolds for all of them. Our results revealed that $\text{GCPN}_{\text{ours}}$ and DFM achieve similar metrics, while the performance of BFM is limited by its preference for molecules with simple structures.

To gain a deeper understanding of the chemical space covered by the molecules generated by these several generative models, we performed UMAP projection on the molecules generated by $\text{GCPN}_{\text{ours}}$ and DFM, as well as 50,000 molecules randomly selected from the ZINC 250k data set. Here, UMAP analysis was performed with the umap-learn Python package [26]. Molecules were represented by their the extended-connectivity fingerprints (ECPF4) fingerprints hashed to 1024 bits. The resulting binary vectors were then reduced to 250 dimensions using principal component analysis before being projected onto two dimensions. The results are illustrated in Figure 1. One can see that both the outcomes given by $\text{GCPN}_{\text{ours}}$ and DFM effectively span the chemical space represented by the training set (i.e., ZINC 250k), indicating their comparable ability to generate diverse molecule structures.

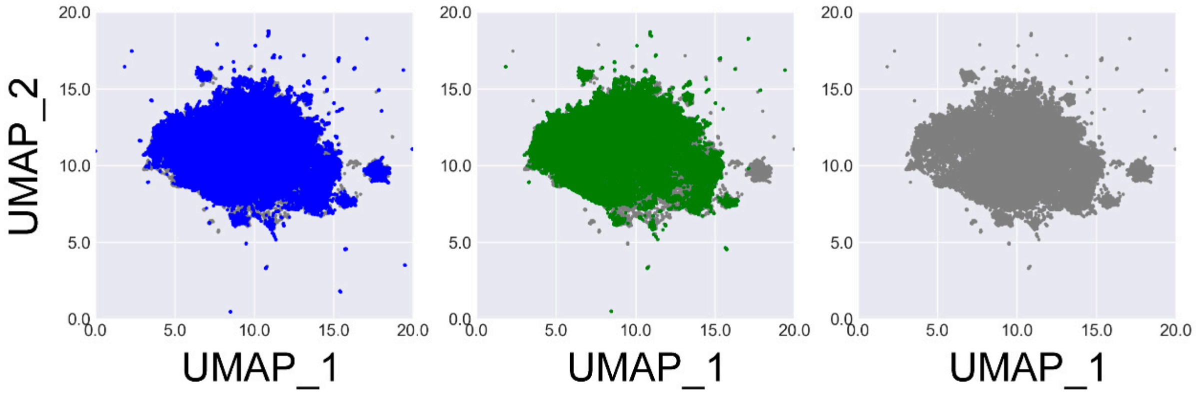


Figure 1. UMAP projection of the ZINC 250k molecules (grey) and those generated by DFM (blue) and GCPN$_{\text{ours}}$ (green), respectively. This plot illustrates the similarity between the chemical spaces covered by different generative models.

2.2. Test Case: De Novo Design with METEOR

As discussed above, GCPN$_{\text{ours}}$ and DFM demonstrated a remarkable advantage over BFM, this test attempted to evaluate the performance of METEOR$_{\text{GCPN}}$ and METEOR$_{\text{DFM}}$ within the realm of reinforcement learning. We then wanted to examine their performance in a real de novo drug design scenario. The objective here was to design ligand molecules targeting glucocerebrosidase (GBA), simultaneously optimizing essential properties including drug-likeness, synthetic accessibility, and binding affinity to the target.

To investigate the effect of multi-objective optimization, we examined the three desired features (i.e., drug-likeness, synthetic accessibility, and binding affinity) of the molecules generated at the initial round and the final round of reinforcement learning (Figure 2). Firstly, a notable improvement in the predicted binding affinity can be observed if comparing the molecules generated by METEOR$_{\text{DFM}}$, METEOR$_{\text{GCPN}}$, and those from ZINC 250k. Regarding the QED value, a significant fraction of the molecules generated by METEOR$_{\text{DFM}}$ and METEOR$_{\text{GCPN}}$ (77.6% and 83.8%, respectively) exceeded the QED threshold of 0.6. Nevertheless, no notable improvement in the QED value was observed after reinforcement learning. This is because the ZINC 250k data set as a whole already exhibits a high level of QED value, leaving very limited room for further improvement. Regarding the SAScore value, the majority of ZINC 250k molecules fall within the range of (1.5, 5.0). After reinforcement learning, SAScore of the generated molecules concentrated at the range of (2.5, 3.5) with a more focused distribution. Furthermore, both models generated fewer molecules that were predicted to be challenging for synthesis as compared to the ZINC 250k molecules. Considering all three features together, the distribution of the unweighted sum of three feature rewards shifted to the right as compared to the distribution of the ZINC 250k molecules. To conclude, both METEOR$_{\text{DFM}}$ and METEOR$_{\text{GCPN}}$ were able to generate novel molecules with improved predicted binding affinity under the constraints of drug-likeness and synthetic accessibility.

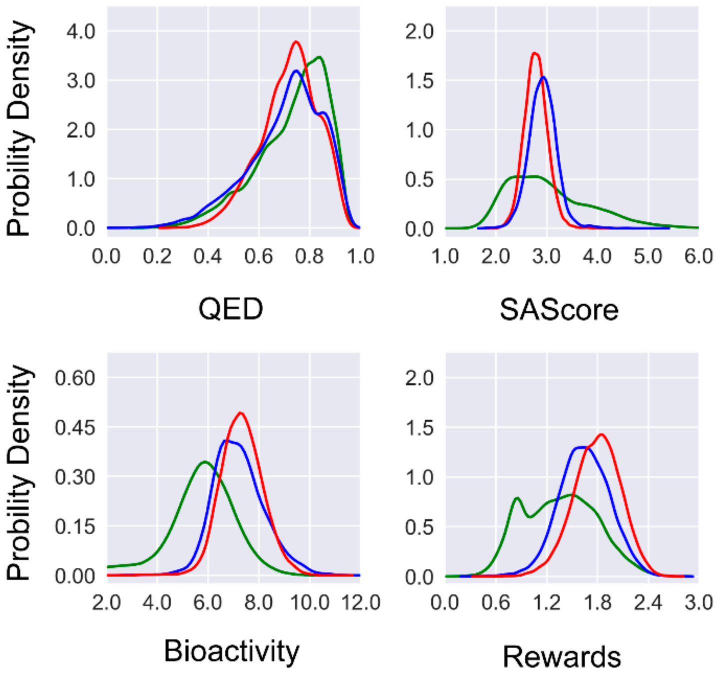


Figure 2. Distribution of three desired features and the unweighted sum of rewards of the molecules generated at the last round of reinforcement learning (red lines: METEOR$_{GCPN}$; blue lines: METEOR$_{DFM}$; green lines: ZINC 250k).

2.3. *METEOR$_{GCPN}$: Has a Larger Action Space as Well as a Higher Learning Efficiency*

Incorporating the depth-first graph traversal algorithms in DFM eliminates the need to decide the starting atom for adding a new bond. This modification reduces the action space of METEOR$_{DFM}$ and theoretically streamlines reinforcement learning. However, the learning curve demonstrated that METEOR$_{GCPN}$ can be trained at a higher level of stability and efficiency than METEOR$_{DFM}$ in reinforcement learning (Figure 3a,b). After 50 rounds of reinforcement learning, METEOR$_{GCPN}$ received a mean total reward of around 1.25, whereas the mean total reward of METEOR$_{DFM}$ at the same point was approximately 0.85. Despite the smaller action space of METEOR$_{DFM}$, the full trajectory for METEOR$_{DFM}$ for generating a molecular graph is roughly twice as long as that of METEOR$_{GCPN}$. This inequality accounts for the different efficiency of METEOR$_{DFM}$ and METEOR$_{GCPN}$. For example, over a three-day period of reinforcement learning, METEOR$_{GCPN}$ generated around 2.7 million molecules across 212 rounds, whereas METEOR$_{DFM}$ generated around 1.8 million molecules over 138 rounds. Given the same amount of training time, the additional training iterations achieved by METEOR$_{GCPN}$ make it possible to uncover molecules with improved properties. Moreover, GCPN's inherent capability of determining when to terminate the graph expansion allows METEOR$_{GCPN}$ to assess the attributes of the existing molecular structure. This capability empowers METEOR$_{GCPN}$ to judiciously halt the expansion of a graph when the current structure exhibits particularly favorable attributes. This explains why METEOR$_{GCPN}$ generated molecules with superior synthetic accessibility in comparison to METEOR$_{DFM.}$

In addition, a notable disparity was observed between the total reward and the unpenalized reward acquired by both METEOR$_{GCPN}$ and METEOR$_{DFM}$ (Figure 3c,d). This gap primarily arose from the property penalty at the early training phase (Figure 3e,f). In METEOR, property penalty (Equation (5)) was the driving force for multi-objective optimization on binding affinity to the protein, drug-likeness, and synthetic accessibility. Computing rewards by a weighted sum across three property rewards reinforced the optimization to be conducted toward all three properties.

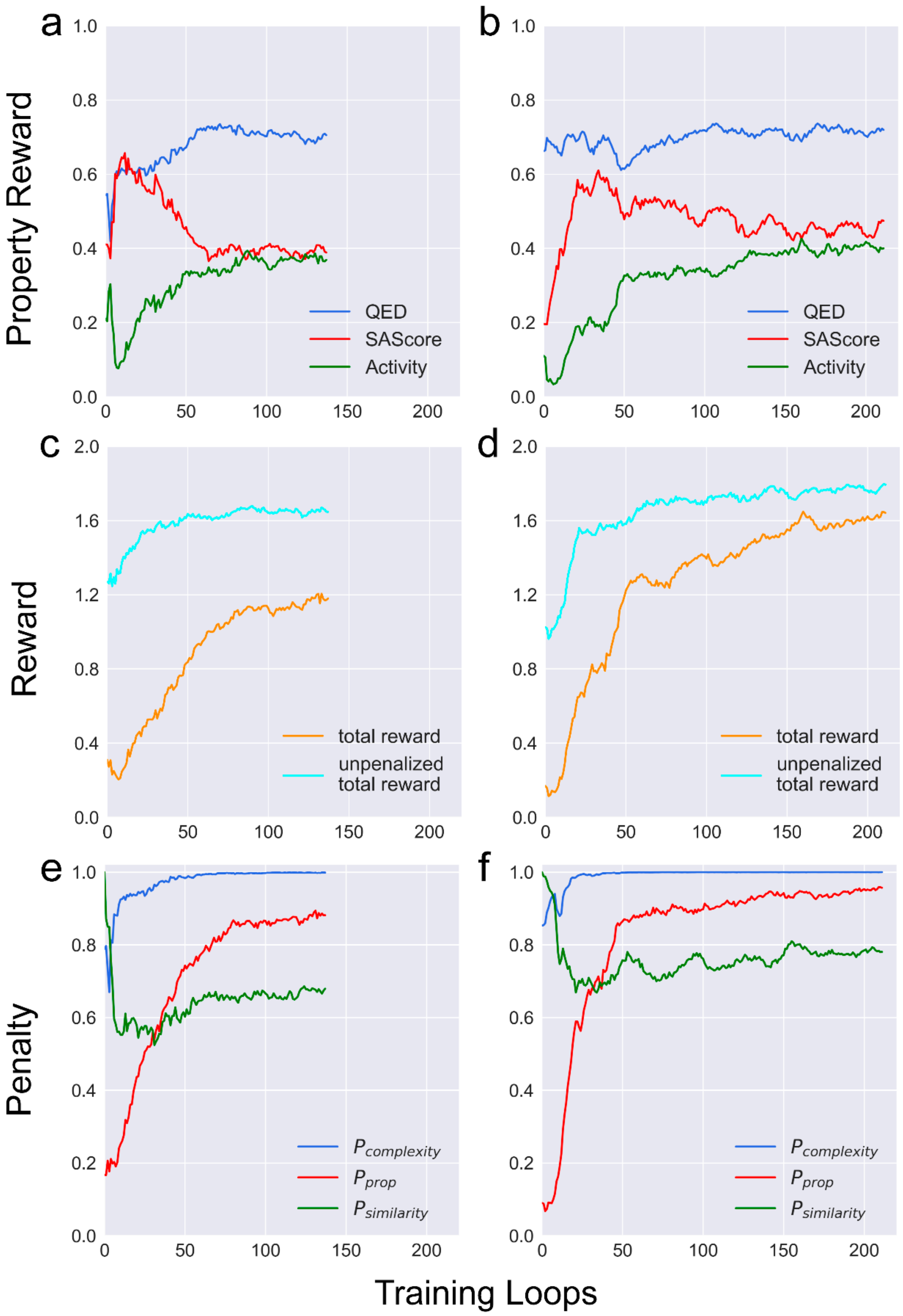


Figure 3. Several key features of METEOR$_{DFM}$ (**left**) and METEOR$_{GCPN}$ (**right**) observed during the reinforcement learning process. (**a**,**b**): Three property rewards; (**c**,**d**): Total and unpenalized rewards; (**e**,**f**): Three penalty factors. Here, each line plots the mean value of a certain feature computed over all molecules generated at each round of roll-out.

The complexity penalty was computed primarily by counting heavy atoms. This penalty was introduced to balance the bias along structure generation, where larger molecules tend to receive higher predicted binding scores by PLANET. Moreover, larger molecules often contain challenging moieties for chemical synthesis, such as chiral centers. The similarity penalty was introduced to encourage METEOR to explore the chemical space preventing it from becoming confined to local maxima. During the initial training rounds, a relatively modest similarity penalty was observed among the molecules generated by both $\text{METEOR}_{\text{DFM}}$ and $\text{METEOR}_{\text{GCPN}}$ due to the presence of limited high-scoring molecules recorded in the memory stack. At the 40th round or so, the influence of similarity penalties became more obvious (Figure 3e,f). Here, both $\text{METEOR}_{\text{DFM}}$ and $\text{METEOR}_{\text{GCPN}}$ were able to explore the full chemical space covered by the ZINC 250k data set throughout the training process without being trapped in certain restricted regions (Figure 4).

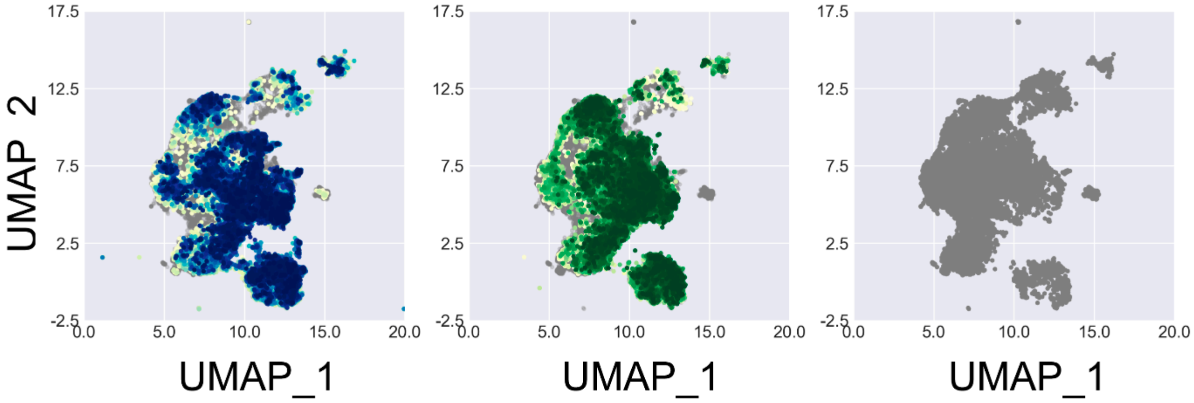


Figure 4. UMAP projections of 50,000 ZINC 250k molecules (grey) and 50,000 molecules generated at certain rounds by $\text{METEOR}_{\text{DFM}}$ (blue) and $\text{METEOR}_{\text{GCPN}}$ (green), respectively. Different rounds are indicated in colors with different shades.

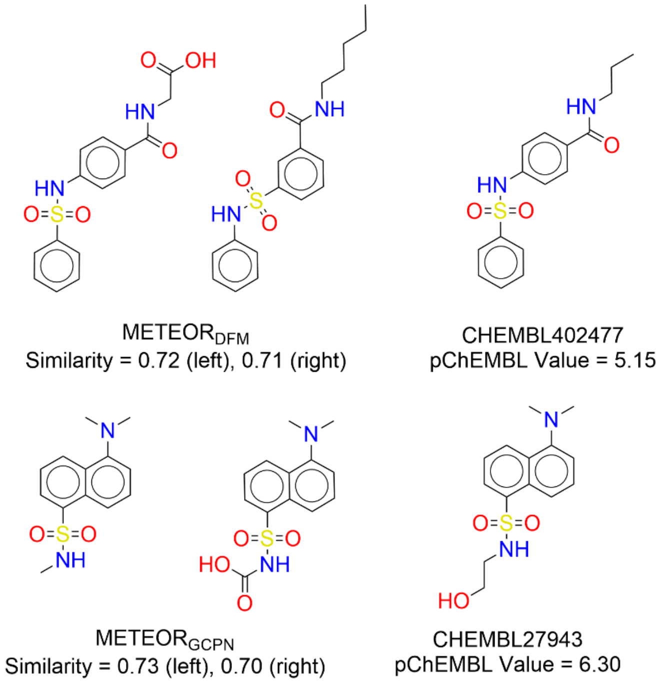
In conclusion, both $\text{METEOR}_{\text{DFM}}$ and $\text{METEOR}_{\text{GCPN}}$ are able to generate molecules with optimized properties. The major distinction between $\text{METEOR}_{\text{DFM}}$ and $\text{METEOR}_{\text{GCPN}}$ lies in their efficiency.

2.4. The Practical Value of METEOR in De Novo Drug Design

In this study, we evaluated the practical value of METEOR in de novo drug design by using GBA as a test case. The quality of the molecules generated by METEOR was reflected by analyzing their similarity to true binders of GBA collected from ChEMBL. If using an ECFP4 Tanimoto coefficient of 0.6 as the threshold, 15 molecules generated by $\text{METEOR}_{\text{DFM}}$ shared similar structures to true binders to GBA. As for $\text{METEOR}_{\text{GCPN}}$, this number was 17. A few such examples are given in Figure 5. One can see that the generated molecules shared an almost identical scaffold as a certain GBA binder. This observation demonstrated that METEOR is able to generate drug-like molecules with potential value.

It should be mentioned though that as a whole, a substantial proportion of the true GBA binders considered in our study have a QED value below 0.5 or SAScores over 3.5 (Figure S3 in the Supporting Information). However, the majority of the molecules generated by METEOR had optimized QED values and SAScores that do not stay at this range (Figure 2). Thus, in this particular test cast, this gap resulted in rather limited matched pairs between the outcomes of METEOR and true GBA binders.

Moreover, we employed the GLIDE module in the Schrödinger software, a widely used conventional molecule docking method, to evaluate the binding affinity of the molecules generated by METEOR with the target protein. Prior to the molecular docking job, the molecules generated during the reinforcement learning process in METEOR were filtered based on the following criteria: (1) an QED value above 0.6, (2) an SAScore lower than 3.0, and (3) a predicted binding affinity value greater than 7.0 (in -log units). The molecules meeting all requirements were then docked into the binding pocket on GBA. To make a comparison, all ZINC 250k molecules were also docked into the binding pocket on GBA through the same protocol.



$\text{METEOR}_{\text{DFM}}$
Similarity = 0.72 (left), 0.71 (right)

CHEMBL402477
pChEMBL Value = 5.15

$\text{METEOR}_{\text{GCPN}}$
Similarity = 0.73 (left), 0.70 (right)

CHEMBL27943
pChEMBL Value = 6.30

Figure 5. Examples of the molecules generated by $\text{METEOR}_{\text{DFM}}$ and $\text{METEOR}_{\text{GCPN}}$ as well as the corresponding true binders of GBA.

As shown in Figure 6, the molecules generated by either $\text{METEOR}_{\text{GCPN}}$ or $\text{METEOR}_{\text{DFM}}$ on average had better GLIDE binding scores than those ZINC 250k molecules, even though even though the optimization of binding affinity in METEOR was guided by a different scoring function PLANET. The 1% percentile of docking scores was −6.20, −6.83, and −6.57 for molecules from ZINC 250k, $\text{METEOR}_{\text{DFM}}$, and $\text{METEOR}_{\text{GCPN}}$, respectively. Note that besides binding affinity to the target protein, the molecules generated by METEOR were also optimized in terms of drug-likeness and synthetic accessibility. Therefore, it is reasonable to expect that more promising active hits can be discovered through application of METEOR rather than a conventional virtual screening of the ZINK 250k data set.

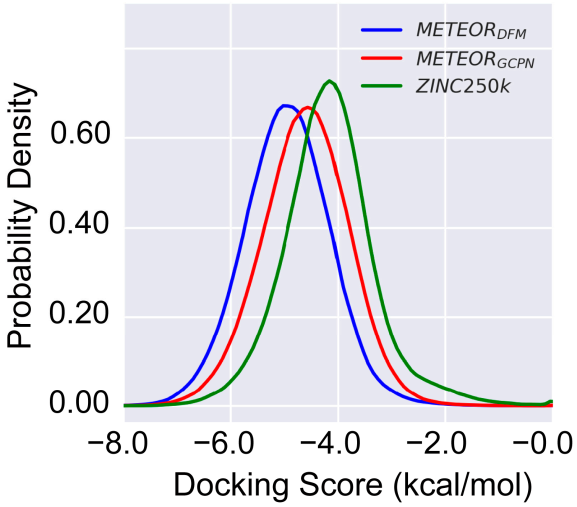


Figure 6. Distribution of the GLIDE docking scores computed for several sets of molecules: ZINC 250k molecules (*n* = 247168, green), $\text{METEOR}_{\text{DFM}}$ molecules (*n* = 279910, blue), and $\text{METEOR}_{\text{GCPN}}$ molecules (*n* = 739130, red).

3. Methods

3.1. The Backend Molecule Generative Models

The GCPN model extends an existing molecular graph by adding new chemical bonds one after another (Figure 7). During this process, four decisions need to be made at each step: (1) determining the starting atom ("focus atom") to which the new bond is added, (2) selecting the end atom of the new bond, (3) specifying the type of the new bond, and (4) deciding whether to terminate graph expansion [8]. The first decision significantly expands the action space of GCPN, leading to numerous possibilities within each existing subgraph. To address this complexity, we introduced two models: DFM and BFM, each employing a distinct graph traversal algorithm (Figure 7). In both models, the "focus atom", defined by the respective graph traversal algorithm, serves as the starting point for adding a new bond. In alignment with DFM and BFM, the final task in GCPN, i.e., determining whether to terminate graph generation, is replaced by marking the current focus atom as "finished". Graph generation terminates when all nodes have been marked.

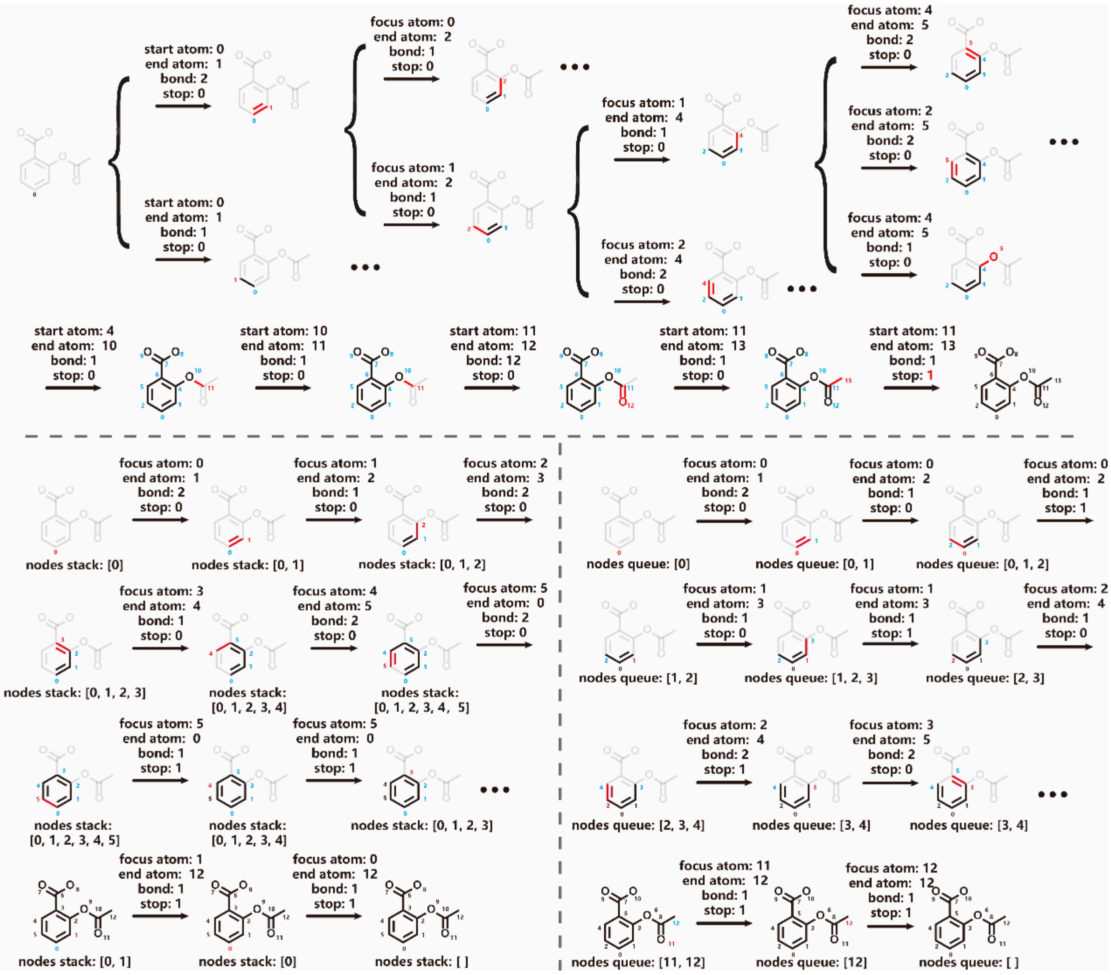

start atom: 0 end atom: 1 bond: 2 stop: 0
start atom: 0 end atom: 1 bond: 1 stop: 0
focus atom: 0 end atom: 2 bond: 1 stop: 0
focus atom: 1 end atom: 2 bond: 1 stop: 0
focus atom: 1 end atom: 4 bond: 1 stop: 0
focus atom: 2 end atom: 4 bond: 2 stop: 0
focus atom: 4 end atom: 5 bond: 2 stop: 0
focus atom: 2 end atom: 5 bond: 2 stop: 0
focus atom: 4 end atom: 5 bond: 1 stop: 0
start atom: 4 end atom: 10 bond: 1 stop: 0
start atom: 10 end atom: 11 bond: 1 stop: 0
start atom: 11 end atom: 12 bond: 12 stop: 0
start atom: 11 end atom: 13 bond: 1 stop: 0
start atom: 11 end atom: 13 bond: 1 stop: 1
focus atom: 0 end atom: 1 bond: 2 stop: 0
focus atom: 1 end atom: 2 bond: 1 stop: 0
focus atom: 2 end atom: 3 bond: 2 stop: 0
nodes stack: [0]
nodes stack: [0, 1]
nodes stack: [0, 1, 2]
focus atom: 3 end atom: 4 bond: 1 stop: 0
focus atom: 4 end atom: 5 bond: 2 stop: 0
focus atom: 5 end atom: 0 bond: 2 stop: 0
nodes stack: [0, 1, 2, 3]
nodes stack: [0, 1, 2, 3, 4]
nodes stack: [0, 1, 2, 3, 4, 5]
focus atom: 5 end atom: 0 bond: 1 stop: 1
focus atom: 5 end atom: 0 bond: 1 stop: 1
nodes stack: [0, 1, 2, 3, 4, 5]
nodes stack: [0, 1, 2, 3, 4]
nodes stack: [0, 1, 2, 3]
focus atom: 1 end atom: 12 bond: 1 stop: 1
focus atom: 0 end atom: 12 bond: 1 stop: 1
nodes stack: [0, 1]
nodes stack: [0]
nodes stack: [ ]
focus atom: 0 end atom: 1 bond: 2 stop: 0
focus atom: 0 end atom: 2 bond: 1 stop: 0
focus atom: 0 end atom: 2 bond: 1 stop: 1
nodes queue: [0]
nodes queue: [0, 1]
nodes queue: [0, 1, 2]
focus atom: 1 end atom: 3 bond: 1 stop: 0
focus atom: 1 end atom: 3 bond: 1 stop: 1
focus atom: 2 end atom: 4 bond: 1 stop: 0
nodes queue: [1, 2]
nodes queue: [1, 2, 3]
nodes queue: [2, 3]
focus atom: 2 end atom: 4 bond: 2 stop: 1
focus atom: 3 end atom: 5 bond: 2 stop: 0
nodes queue: [2, 3, 4]
nodes queue: [3, 4]
nodes queue: [3, 4]
focus atom: 11 end atom: 12 bond: 1 stop: 1
focus atom: 12 end atom: 12 bond: 1 stop: 1
nodes queue: [11, 12]
nodes queue: [12]
nodes queue: [ ]

Figure 7. Illustration of graph generation process for the aspirin molecule by GCPN (**top**), DFM (**bottom left**, in a depth-first manner), and BFM (**bottom right**, in a breadth-first manner). At each step, the end atoms of new bonds are marked in red, possible focus atoms are marked in blue, and "finished atoms" are depicted in black.

Individual atom nodes were encoded using vectors with a dimension of 20, consisting of one-hot encoded element type, atom degree, and membership in rings of varying sizes from 3 to 7. These initial vectors were then embedded into a latent space (h_0) with a size of 64 dimensions. To extract features from the input graphs, we utilized the graph convolution network (GCN) architecture [27–29]. The entire molecular graph was partitioned into three distinct subgraphs based on different bond orders. One module with three separate graph convolution layers with a hidden size of 64, each with learnable parameters W_i^l, were employed on the three subgraphs, as denoted in Equation (1).

$$h^l = \sum_{i=1}^{b}\left(ReLU\left(\widetilde{D}_i^{-\frac{1}{2}} \widetilde{A}_{\underset{i}{:}} \widetilde{D}_i^{-\frac{1}{2}} h^{(l-1)} W_i^l \right)\right), \mathrm{b} \in \{1,2,3\} \tag{1}$$

$A_{\underset{i}{:}}$ is the ith slice bond-conditioned adjacent matrix, $\widetilde{A}_{\underset{i}{:}} = A_{\underset{i}{:}} + I$; $\widetilde{D}_i$ is the ith slice bond-conditioned degree matrix with self-loop.

In our implementation, we utilized three such modules to extract the underlying features from the molecular graphs. The extracted latent features were subsequently utilized to make informed decisions within the model. The configuration of task layers in GCPN remained consistent with the original literature [8]. In the case of DFM and BFM, when selecting ending atom for the chemical bond to be added, nodes marked as "finished" were overlooked. The probability of each action $P(a_t)$ was calculated as shown in Equation (2):

$$P(a_t) = \left(1 - \mathbb{I}\left(a_t^{\text{finish}}\right)\right)\prod\nolimits_j P\left(a_t^j\right) + \mathbb{I}\left(a_t^{\text{finish}}\right) P\left(a_t^{\text{finish}}\right) \tag{2}$$

Each action step a_t is composed of three sub-tasks a_t^j, i.e., selecting the end atom of the new bond, specifying type of the new bond, and whether to mark the focus atom as "finished". $\mathbb{I}\left(a_t^{\text{finish}}\right)$ equals 0 if the focus atom is not decided to be marked, else $\mathbb{I}\left(a_t^{\text{finish}}\right)$ equals 1.

To enhance stability and performance in reinforcement learning, a commonly employed strategy involves pre-training a generative model using an established compound database [30]. In our study, we employed the widely-used ZINC 250k data set, comprising structurally diverse "drug-like" molecules that have been synthesized in reality. Structures of these molecules were examined to eliminate those containing rings with eight or more members. The remaining molecules were considered as the ground truth and served as expert training data. For GCPN pre-training, a randomly sampled connected subgraph G′ from a molecule graph G was viewed as the state s_t. Any action a_t added an atom or a bond in G but not in G′ could be viewed as an expert action during the trajectory of generating a ground-truth molecule. The training objective was to maximize the possibility $P(a_t)$ of GCPN to take expert action a_t at state s_t. This training approach was similar to that previously reported [8]. For both DFM and BFM, the molecular structures from the filtered ZINC 250k data set were transformed into expert trajectories by randomly selecting a starting node and traversing the graphs in a depth-first or breadth-first manner, respectively. The resulting expert actions a_t consisting of the trajectories were collected for pre-training DFM and BFM. The objective in expert training for all models can be expressed as shown in Equation (3):

$$L^{\text{expert}}(\theta) = -\frac{1}{T}\sum_{t}^{T} \log P(a_t) \tag{3}$$

The Adam optimizer with a learning rate of 0.0001 was applied. After 1,000,000 training steps, GCPN, DFM, and BFM with converged loss were obtained as pre-trained generative models.

3.2. The Molecule Generation Environment

Within the context of molecular graph generation under a reinforcement learning framework, the molecule generation environment plays two essential roles, i.e., state transition dynamics and reward assignment.

3.2.1. State Transition Dynamics

The molecule generation environment plays a pivotal role in executing the actions taken by the agent, ensuring adherence to specified rules. One fundamental rule incorporated into the environment is the valency check, preventing actions that exceed an atom's maximal valency [8]. It is noteworthy that substructures adhering to the basic valency rule may not be "drug-like". Therefore, the environment in our model detects and then filters out the following "non-drug-like" substructures: (a) cumulative alkenes and peroxyl bonds; (b) double or triple bonds in a three- or four-membered ring; (c) bridged ring formed with aromatic rings; and (d) large rings with eight or more members, as detected in the smallest set of smallest rings in a molecular graph. Substructure detection is performed after each agent action through SMARTS matching (see Figure S4 in the Supporting Information). Only actions that pass both the valency check and substructure examination will be adopted by the environment to update the current molecule subgraph. Note that implementation of the above chemical rules reflects the knowledge of "drug-likeness" accumulated in the literature [31–33]. There are of course different perceptions of "drug-likeness", but the several rules listed above are relatively straightforward to be encoded in a computer program. In particular, macrocyclic structures are not allowed in our model, although some marketed drugs do consist of such structures [34]. From a practical view, macrocyclic structures are normally introduced at the stage of lead optimization to impose conformational constraints. Considering that our model will be employed primarily as an "idea generator" at the stage of lead discovery, ignoring macrocyclic structures is an acceptable trade-off for the sake of technical convenience.

3.2.2. Reward Assignment

The behavior of agents is steered by the rewards from the molecule generation environment, which can be categorized into two components: step reward and final reward. A zero-step reward is assigned to each step, except for two specific actions: (a) When a new ring is formed, a small step reward of 0.02 is assigned to encourage ring formation. (b) When an improper action is canceled by the molecular generation environment, a step reward of -0.2 is given to discourage such actions. The step reward serves to guide the agent's behavior and reduce the occurrence of improper actions. The final reward comprises several domain-specific rewards assigned based on different properties, including drug-likeness, synthetic accessibility, and predicted bio-activity. The final reward is calculated as the weighted sum of these rewards, further adjusted by a penalty factor. Reward functions related to specific properties utilize a linear scaling function that maps values between a lower bound and an upper bound, as described in Equation (4):

$$R_{prop} = \begin{cases} 1.0 & S_{prop} \geq S_{prop}^{high}\ ; \\ \frac{S_{prop}-S_{prop}^{low}}{S_{prop}^{high}-S_{prop}^{low}}, & S_{prop}^{low} < S_{prop} < S_{prop}^{high}; \\ 0.0 & S_{prop} \leq S_{prop}^{low}\ . \end{cases} \tag{4}$$

This type of function is chosen based on the assumption that it is not necessary to optimize certain properties beyond desired ranges. For example, it is not necessary to further optimize the synthetic accessibility of a molecule with an SAScore lower than 2.0 because it is already good enough at this level.

Generated molecules are evaluated by the following three properties:

(a) Drug-likeness of a molecule is assessed by the QED index originally proposed by Hopkins et al. [33]. This index has a range (0.0, 1.0).

(b) Synthetic accessibility of a molecule is evaluated by SAScore, which has a range [1.0, 10.0]. SAScore is a rule-based tool for estimating synthetic accessibility, and its output is determined by the summation of fragment scores and a complexity penalty [35].

(c) Binding affinity to the target protein is predicted by PLANET, a graph neural network model developed in our group [25]. PLANET operates on two-dimensional molecular graphs as inputs and thus skips the exhaust molecular docking process. Its ultra-fast speed is suitable for processing generated molecules in a large number.

A penalty factor is also implemented to influence the agent model's behavior by scaling the sum of property rewards. This factor is determined based on three aspects:

(a) Complexity penalty ($P_{\text{complexity}}$). The complexity penalty is assigned based on the number of heavy atoms in the designed molecule, defined as a linear scaling function akin to Equation (4). The lower and upper bounds for the number of heavy atoms are set to 10 and 40, respectively. Additionally, for molecules with more than two chiral centers, a penalty factor of 0.5 will multiply $P_{\text{complexity}}$.

(b) Property penalty (P_{prop}). Since the reward is the sum of three property rewards, it is possible for an agent to receive a high reward from a molecule that possesses two excellent properties but one extremely poor property. The property penalty is applied as follows (Equation (5)):

$$P_{\text{prop}} = \prod\nolimits_{prop} \min\left(1.0, R_{prop}/0.2\right) \tag{5}$$

(c) Similarity penalty ($P_{\text{similarity}}$). Agents trained in reinforcement learning tend to generate highly-scored molecules. However, once a local maximum is reached, agents often struggle to explore other areas, leading to a phenomenon known as "policy collapse". Inspired by the work of Blaschke et al. [14], we devised a similarity penalty to encourage agents not only to focus on specific favorable regions in the chemical space yielding high scores but also to explore various areas within the space. Our algorithm for calculating the similarity penalty differs from that of Blaschke though. For example, all halogen atoms are ignored here to prevent our model from generating molecular structures with differences merely in the number and position of halogen atoms. This is important since at the stage of lead discovery, sufficient diversity in the structural scaffold is much desired, where terminal halogen atoms are not part of a structural scaffold. In fact, halogen atoms are often added to optimize bioactivity at a later stage of drug discovery. Subsequently, a mapping between the current molecule and those generated before the preceding twenty rounds of roll-out is performed. If a successful mapping is found, a zero-penalty factor is assigned. Molecules passing this mapping step proceed to subsequent similarity calculation. A stack is used to retain favorable molecules, that is, those generated over the preceding 10 rounds of roll-out with a final property reward surpassing 70% of the possible maximum. Tanimoto similarity coefficients between the ECFP4 of the transformed molecule and all stored high-quality molecules are calculated. $P_{\text{similarity}}$ is determined based on the maximal Tanimoto similarity coefficient, as outlined in Equation (6):

$$P_{\text{similarity}} \begin{cases} 0.0 & \text{Success Mapping } \textbf{OR} \text{ Tanimoto} \geq 0.7\,; \\ 1 - (\text{Tanimoto} - 0.4)/0.3, & 0.4 < \text{Tanimoto} < 0.7\,; \\ 1.0 & \text{Tanimoto} \leq 0.4\,. \end{cases} \tag{6}$$

The final reward (R_{final}) is the weighted sum of all property rewards scaled by overall penalty (Equation (7)):

$$R_{\text{final}} = \sum^{i} \alpha_i R_i \times \prod^{j} P_j \tag{7}$$

Here, *i* and *j* denote for different types of molecular properties and penalty factors, respectively.

3.2.3. Reinforcement Learning

Policy gradient-based methods are widely adopted in reinforcement learning. In our model, Proximal Policy Optimization (PPO) is adopted [36]. The learning objective can be written as Equation (8):

$$L_{PPO}^{\theta^k}(\theta) = -\sum_{(s_t,a_t)} min\left(\frac{P_\theta(a_t|s_t)}{P_\theta^k(a_t|s_t)} A^{\theta^k}(a_t,s_t), clip\left(\frac{P_\theta(a_t|s_t)}{P_\theta^k(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon\right) A^{\theta^k}(a_t,s_t)\right)$$

$$A^{\theta^k}(a_t,s_t) = \sum_{t=t'}^{T_n} \gamma^{T_n - t'} R_{\text{final}} + R_{\text{step}} - b \tag{8}$$

In the objective function, γ represents the discount factor, and its value is experimentally set to 0.98. The clip value, denoted as ε, is set to 0.1. The superscript k denotes for the generative model obtained after the last round of training. The estimated advantage function $A^{\theta^k}(a_t,s_t)$ incorporates a learnable value function b. The value function takes the same molecular graph embedding obtained from the GCN layers and maps it to a scalar representing the estimated expected reward. The probability of an action taken by the generative model with parameter θ under state s_t, denoted as $P_\theta(a_t|s_t)$, is calculated using Equation (2).

3.3. Performance Evaluation

3.3.1. Evaluation in Terms of Generating Valid Molecular Structures

We assessed the performance of several pre-trained generative models, including the GCPN$_{\text{origin}}$ (only enabling valency check during graph generation), the GCPN$_{\text{ours}}$, DFM, and BFM (all three utilizing full substructure check, including valency check and improper substructure detection, see Figure S4 in the Supporting Information). Each pre-trained model was assigned the task of generating 50,000 molecules for evaluating validity, uniqueness, and novelty as follows:

$$\text{Validity} = \frac{\text{Number of valid graphs}}{\text{Number of generated graphs}}$$

$$\text{Uniqueness} = \frac{\text{Number of unique and valid graphs}}{\text{Number of valid graphs}}$$

$$\text{Novelty} = \frac{\text{Number of unique and valid graphs not in the training set}}{\text{Number of unique and valid graphs}}$$

Valid graphs were typically measured with respect to valency and bonds using RDKit's molecular structure parser.

3.3.2. Evaluation in Terms of Generating Useful Hits on a Specific Target Protein

To evaluate the effectiveness of METEOR in generating useful hits in a de novo drug design scenario, we chose GBA as the target protein, which is included in the popular LIT-PCBA benchmark for testing virtual screening methods [37]. The crystal structure of GBA used in our test is obtained from the Protein Data Bank (PDB entry 2V3D) [38]. Two pre-trained generative models, namely, GCPN$_{\text{ours}}$ and DFM, served as backends of METEOR (denoted as METEOR$_{\text{GCPN}}$ and METEOR$_{\text{DFM}}$, respectively, hereafter). Major adjustable parameters in these models are in Equation (4), where the lower bound of QED, SAScore, and binding affinity was set to 0.2, 3.5, and 5.5, respectively, and the upper bound was set to 0.8, 2.0, and 8.5, respectively.

Our test was performed on a server equipped with two NVIDIA GeForce 2080Ti GPU cards (11 GB memory), two Intel(R) Xeon(R) Silver 4210 CPUs @ 2.20 GHz, and 128 GB of

RAM. After three days of reinforcement learning with 10 parallel processes, all generated molecules were assessed in two aspects: Firstly, a total of 452 true binders of GBA were curated from ChEMBL, which were identified by a "Target ChEMBL ID" of CHEMBL2179 and a "pChEMBL value" greater than 5.0 (for example, K_d or Ki value < 10 μM). Pairs of molecules with an ECFP4 Tanimoto coefficient over 0.6 were defined as similar. The total number of molecules generated by METEOR that were similar to true binders to GBA was counted and analyzed. Secondly, the molecules generated by METEOR, filtered based on QED value, SAScore, and binding affinity, were docked into the binding pocket of GBA by using GLIDE in the standard precision (SP) mode in the Schrödinger software. To make a comparison, the molecules in the ZINC 250k data set were docked into the binding pocket of GBA following the same protocol.

4. Conclusions

In this work, we have developed a deep learning mode, called METEOR, for potential application in de novo drug design. Compared to many other generative models already described in the literature, METEOR has several distinct technical features.

Firstly, the backend agent of METEOR is based on the well-established GCPN model. We have evaluated several graph traversal algorithms within a reinforcement learning framework. Our findings indicate that depth-first graph generation (DFM) outperforms breadth-first graph generation (BFM). Its outcomes closely align with those of the original GCPN model in terms of validity, uniqueness, and novelty. This observation supports the potential value of both $\text{METEOR}_{\text{GCPN}}$ and $\text{METEOR}_{\text{DFM}}$ in de novo drug design. As demonstrated in the test case of GBA, without prior knowledge of true binders, both models are able to generate molecules with superior properties compared to those in the ZINC 250k data set.

Secondly, in order to ensure the overall validity of the generated molecular structures, we have implemented a set of chemical rules in METEOR to eliminate undesired substructures. In fact, if these rules are not enabled, a significant portion (~40%) of the generated molecule structures would be undesirable. This demonstrates the importance of integrating chemical knowledge into molecular structure generation, which has become a new trend in this field (for example, see a new generative model published recently [39]).

Last, and very importantly, unlike many other generative models that focus on a single objective, METEOR is designed to generate molecules with optimized traits regarding binding affinity, drug-likeness, and synthetic accessibility. These several properties are all indispensable for a successful candidate in the early phase of drug discovery. This makes METEOR better suited for practical applications to drug discovery.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/molecules30010018/s1, Figure S1: Examples of valid but unwanted molecular structures; Figure S2: Examples of molecules with "ring closure" issue; Figure S3: Distribution of QED, SAScore and PLANET binding scores of the known GBA binders collected from ChEMBL; Figure S4: Valency check and improper substructure detection implemented in different generative models.

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

Funding: This study was financially supported by the National Natural Science Foundation of China (Grant No. 81725022 and 82173739 to R. Wang), the Ministry of Science and Technology of China (National Key Research Program, Grant No.2023YFF1205102 to Y. Li), and the Shanghai Natural Science Foundation (Grant No. 24ZR1413800 to Y. Li).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Source code of METEOR, as well as user manual and demo examples, are available at https://github.com/ComputArtCMCG/METEOR (accessed on 12 December 2024).

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

References

1. Sterling, T.; Irwin, J.J. ZINC 15—Ligand discovery for everyone. *J. Chem. Inf. Model.* **2015**, *55*, 2324–2337. [CrossRef] [PubMed]
2. Kirkpatrick, P.; Ellis, C. Chemical space. *Nature* **2004**, *432*, 823. [CrossRef]
3. Vanhaelen, Q.; Lin, Y.-C.; Zhavoronkov, A. The advent of generative chemistry. *ACS Med. Chem. Lett.* **2020**, *11*, 1496–1505. [CrossRef] [PubMed]
4. Sousa, T.; Correia, J.; Pereira, V.; Rocha, M. Generative deep learning for targeted compound design. *J. Chem. Inf. Model.* **2021**, *61*, 5343–5361. [CrossRef]
5. Atance, S.R.; Diez, J.V.; Engkvist, O.; Olsson, S.; Mercado, R. De novo drug design using reinforcement learning with graph-based deep generative models. *J. Chem. Inf. Model.* **2022**, *62*, 4863–4872. [CrossRef]
6. Arus-Pous, J.; Johansson, S.V.; Prykhodko, O.; Bjerrum, E.J.; Tyrchan, C.; Reymond, J.L.; Chen, H.; Engkvist, O. Randomized SMILES strings improve the quality of molecular generative models. *J. Cheminform.* **2019**, *11*, 71. [CrossRef]
7. Sridharan, B.; Goel, M.; Priyakumar, U.D. Modern machine learning for tackling inverse problems in chemistry: Molecular design to realization. *Chem. Commun.* **2022**, *58*, 5316–5331. [CrossRef]
8. You, J.X.; Liu, B.W.; Ying, R.; Pande, V.; Leskovec, J. Graph convolutional policy network for goal-directed molecular graph generation. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 6410–6421.
9. Mariya, P.; Mykhailo, S.; Junier, O.; Olexandr, I. MolecularRNN: Generating realistic molecular graphs with optimized properties. *arXiv* **2019**. [CrossRef]
10. Papadopoulos, K.; Giblin, K.A.; Janet, J.P.; Patronov, A.; Engkvist, O. De novo design with deep generative models based on 3D similarity scoring. *Bioorganic Med. Chem.* **2021**, *44*, 116308. [CrossRef]
11. Korshunova, M.; Huang, N.; Capuzzi, S.; Radchenko, D.S.; Savych, O.; Moroz, Y.S.; Wells, C.I.; Willson, T.M.; Tropsha, A.; Isayev, O. Generative and reinforcement learning approaches for the automated de novo design of bioactive compounds. *Commun. Chem.* **2022**, *5*, 129. [CrossRef] [PubMed]
12. Sicho, M.; Luukkonen, S.; van den Maagdenberg, H.W.; Schoenmaker, L.; Beiquignon, O.J.M.; van Westen, G.J.P. DrugEx: Deep learning models and tools for exploration of drug-like chemical space. *J. Chem. Inf. Model.* **2023**, *63*, 3629–3636. [CrossRef] [PubMed]
13. Zhang, W.; Zhang, K.; Huang, J. A simple way to incorporate target structural information in molecular generative models. *J. Chem. Inf. Model.* **2023**, *63*, 3719–3730. [CrossRef] [PubMed]
14. Blaschke, T.; Engkvist, O.; Bajorath, J.; Chen, H.M. Memory-assisted reinforcement learning for diverse molecular de novo design. *J. Cheminform.* **2020**, *12*, 68. [CrossRef]
15. Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H.M. Molecular de-novo design through deep reinforcement learning. *J. Cheminform.* **2017**, *9*, 48. [CrossRef]
16. Zhavoronkov, A.; Ivanenkov, Y.A.; Aliper, A.; Veselov, M.S.; Aladinskiy, V.A.; Aladinskaya, A.V.; Terentiev, V.A.; Polykovskiy, D.A.; Kuznetsov, M.D.; Asadulaev, A.; et al. Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nat. Biotechnol.* **2019**, *37*, 1038–1040. [CrossRef]
17. Mokaya, M.; Imrie, F.; van Hoorn, W.P.; Kalisz, A.; Bradley, A.R.; Deane, C.M. Testing the limits of SMILES-based de novo molecular generation with curriculum and deep reinforcement learning. *Nat. Mach. Intell.* **2023**, *5*, 386–394. [CrossRef]
18. Mazuz, E.; Shtar, G.; Shapira, B.; Rokach, L. Molecule generation using transformers and policy gradient reinforcement learning. *Sci. Rep.* **2023**, *13*, 8799. [CrossRef]
19. Bilodeau, C.; Jin, W.; Jaakkola, T.; Barzilay, R.; Jensen, K.F. Generative models for molecular discovery: Recent advances and challenges. *WIREs Comput. Mol. Sci.* **2022**, *12*, e1608. [CrossRef]
20. Tan, Y.; Dai, L.; Huang, W.; Guo, Y.; Zheng, S.; Lei, J.; Chen, H.; Yang, Y. DRlinker: Deep Reinforcement Learning for Optimization in Fragment Linking Design. *J. Chem. Inf. Model.* **2022**, *62*, 5907–5917. [CrossRef]
21. Wengong, J.; Regina, B.; Tommi, J. Multi-Objective Molecule Generation using Interpretable Substructures. *arXiv* **2020**. [CrossRef]

22. Janet, J.P.; Ramesh, S.; Duan, C.; Kulik, H.J. Accurate Multiobjective Design in a Space of Millions of Transition Metal Complexes with Neural-Network-Driven Efficient Global Optimization. *ACS Cent. Sci.* **2020**, *6*, 513–524. [CrossRef] [PubMed]
23. Kneiding, H.; Nova, A.; Balcells, D. Directional multiobjective optimization of metal complexes at the billion-system scale. *Nat. Comput. Sci.* **2024**, *4*, 263–273. [CrossRef] [PubMed]
24. Mercado, R.; Bjerrum, E.J.; Engkvist, O. Exploring graph traversal algorithms in graph-based molecular generation. *J. Chem. Inf. Model.* **2022**, *62*, 2093–2100. [CrossRef] [PubMed]
25. Zhang, X.; Gao, H.; Wang, H.; Chen, Z.; Zhang, Z.; Chen, X.; Li, Y.; Qi, Y.; Wang, R. PLANET: A multi-objective graph neural network model for protein–ligand binding affinity prediction. *J. Chem. Inf. Model.* **2023**, *64*, 2205–2220. [CrossRef]
26. McInnes, L.; Healy, J.; Melville, J. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv* **2018**, arXiv:1802.03426.
27. Duvenaud, D.; Maclaurin, D.; Aguilera-Iparraguirre, J.; Gómez-Bombarelli, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R.P. Convolutional networks on graphs for learning molecular fingerprints. *Adv. Neural Inf. Process. Syst.* **2015**, *2*, 2224–2232.
28. Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; Riley, P. Molecular graph convolutions: Moving beyond fingerprints. *J. Comput. Aided Mol. Des.* **2016**, *30*, 595–608. [CrossRef]
29. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**. [CrossRef]
30. Levine, S.; Koltun, V. Guided policy search. *Proc. Mach. Learn.* **2013**, *28*, 1–9.
31. Leeson, P.; Springthorpe, B. The influence of drug-like concepts on decision-making in medicinal chemistry. *Nat. Rev. Drug Discov.* **2007**, *6*, 881–890. [CrossRef] [PubMed]
32. Ursu, O.; Rayan, A.; Goldblum, A.; Oprea, T.I. Understanding drug-likeness. *WIREs Comput. Mol. Sci.* **2011**, *1*, 760–781. [CrossRef]
33. Bickerton, G.R.; Paolini, G.V.; Besnard, J.; Muresan, S.; Hopkins, A.L. Quantifying the chemical beauty of drugs. *Nat. Chem.* **2012**, *4*, 90–98. [CrossRef]
34. Jimenez, D.G.; Poongavanam, V.; Kihlberg, J. Macrocycles in Drug Discovery: Learning from the Past for the Future. *J. Med. Chem.* **2023**, *66*, 5377–5396. [CrossRef]
35. Ertl, P.; Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *J. Cheminform.* **2009**, *1*, 8. [CrossRef]
36. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**. [CrossRef]
37. Tran-Nguyen, V.-K.; Jacquemard, C.; Rognan, D. LIT-PCBA: An unbiased data set for machine learning and virtual screening. *J. Chem. Inf. Model.* **2020**, *60*, 4263–4273. [CrossRef]
38. Brumshtein, B.; Greenblatt, H.M.; Butters, T.D.; Shaaltiel, Y.; Aviezer, D.; Silman, I.; Futerman, A.H.; Sussman, J.L. Crystal structures of complexes of N-butyl- and N-nonyl-deoxynojirimycin bound to acid β-glucosidase. *J. Biol. Chem.* **2007**, *282*, 29052–29058. [CrossRef]
39. Jiang, Y.; Zhang, G.; You, J.; Zhang, H.; Yao, R.; Xie, H.; Zhang, L.; Xia, Z.; Dai, M.; Wu, Y.; et al. PocketFlow is a data-and-knowledge-driven structure-based molecular generative model. *Nat. Mach. Intell.* **2024**, *6*, 326–337. [CrossRef]

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