MolFinder: an efficient global molecular property optimization and search algorithm using SMILES

Yongbeom Kwon[†,‡] and Juyong Lee[*,†]

†*Division of Chemistry and Biochemistry, Department of Chemistry, Kangwon National University, Chuncheon, 24341, Republic of Korea*

‡*Current address: Arontier co., 241, Gangnam-daero, Seoul, 06735, Republic of Korea*

E-mail: juyong.lee@kangwon.ac.kr

Abstract

Here, we introduce a new molecule optimization method, MolFinder, based on an efficient global optimization algorithm, the conformational space annealing algorithm, and the SMILES representation. MolFinder finds diverse molecules with desired properties efficiently without any training and a large molecular database. Compared with recently proposed reinforcement-learning-based molecule optimization algorithms, MolFinder consistently outperforms in terms of both the optimization of a given target property and the generation of a set of diverse and novel molecules. The efficiency of MolFinder demonstrates that combinatorial optimization using the SMILES representation is a promising approach for molecule optimization, which has not been well investigated despite its simplicity. We believe that our results shed light on new possibilities for advances in molecule optimization methods.

Introduction

An inverse molecular design approach, finding valuable molecules with desired properties for a given application, is drawing attention from chemists recently. Conventional molecular design approaches find novel molecules by perturbing known molecules using experienced chemists' intuition. For validation, the designed molecules should be synthesized and tested through experiments. This whole procedure requires considerable time and resources to complete, which retards the development of novel valuable molecules. On the other hand, the inverse molecular design determines the desired properties or properties first and then searches/generates candidate molecules that are assumed to have desired properties.[1,2] With the help of the recent development of artificial intelligence (AI)/machine learning (ML), the inverse molecular design is expected to accelerate the discovery of novel molecules in various fields including the pharmaceutical industry.[3]

Various inverse molecular design methods using AI have been actively developed recently.[4] The most commonly used strategy for molecular design is to use the SMILES representation, which is a character-based linear notation in which the structure of the molecule is considered.[5] In other words, the SMILES string contains information about the structure and stereochemistry of a molecule and the presence of electric charges. The following are a few examples of ML-based molecule generation models. First, various methods have been developed based on the variational autoencoder (VAE) algorithm.[6–8] VAE-based approaches convert input SMILES strings or molecular graphs into multi-dimensional vectors on a latent space based on their similarities and physicochemical properties. It is also shown that molecular transformations are possible by vector transformation on the numerical chemical space. Second, many methods that generate novel SMILES strings have been suggested based on the recurrent neural network (RNN) models.[9,10] In these methods, RNN-models are trained to learn the syntax of the SMILES representation from a large set of molecule database. After initial training, the models are used to generate novel SMILES strings. Generally, RNN-based methods have two inherent limitations. First, not all generated SMILES strings

are valid; some generated strings violate the syntax of SMILES. Second, generated SMILES strings may overlap with those in the training set.

Efforts are being made to create the models that generate molecules with desired properties using the idea of reinforcement-learning (RL).[10–14] RL is an area of ML that aims to obtain the best of the selectable behaviors based on the current environment. As an example, the ReLeaSE algorithm[11] performed RL with a SMILES generating model using stacked-RNN cells[15] trained with known chemical databases. ReLeaSE was shown to generate molecules with desired physicochemical properties and was used to design possible strong binders of the JAK2 proteins. Another RL-based molecular design model is Molecule DQN (MolDQN),[12] which is based on the Deep Q-Networks (DQN) algorithm.[16] MolDQN uses predefined molecular variation operations to modify existing molecules into new molecules suitable for their purposes. Together with the VAE approach, RL-VAE models that improve the fitness of molecules produced by VAE via RL have been suggested.[17,18] More comprehensive reviews of various ML-based molecular generation and optimization methods are given in detail in recent papers.[4,19–21]

The above ML-based models must be trained using existing molecular libraries such as ZINC,[22] ChEMBL,[23] and PubChem.[24] One potential limitation of ML-based approaches is that the results of these models heavily depend on training data. In other words, these models may be difficult to generate novel molecules that are highly dissimilar to the molecules seen during training. For example, in the case of the VAE model, the latent multi-dimensional space is constructed based on the similarities between input molecules, which guarantees good interpolation between known molecules. However, it is still not clear whether extrapolation on the latent space will yield valid molecules. In summary, ML-based models suffer from strong training data dependence, which may bias the quality and quantity of generated molecules.

In addition to recent ML-based approaches, various genetic algorithm (GA)-based molecular property optimization algorithms have been developed.[25–33] The main advantage of GA-

based algorithms is that they do not require a large amount of molecule data relevant to a given optimization task because they search novel molecules in a combinatorial and stochastic way. Also, they do not need to train a molecule generator, which takes considerable computational time and resources. Most existing GA-based molecular optimization algorithms are based on the graph representation of a molecule. In recent studies, they showed competitive, sometimes better, performance compared to ML-based methods in generating novel molecules with desired properties.[25,26,28,29] A GA-based method using the graph representation requires careful design of crossover and/or mutation operations of graphs, which may bias the direction and extension of chemical space search. In addition, the design of any arbitrary operation may be limited because generally it is tightly coupled with the molecular manipulation functionality of underlying cheminformatics libraries, such as RDKit.[34] Alternative to graph-based approaches, Yoshikawa et al. proposed a GA method by converting a SMILES string into a 200-dimensional integer array based on a certain grammar.[30] However, interestingly, performing GA using the SMILES representation itself has not been well investigated despite its simplicity and computational efficiency.[31,33] The approach has been considered less efficient than the graph-based approaches.[28,29,32,33]

In this study, we propose the *MolFinder* method, which is a new molecular design algorithm using the conformational space annealing (CSA) algorithm,[35] a class of an evolutionary algorithm. Unlike most existing graph-based GA methods, MolFinder performs a highly efficient global optimization of molecular properties using the SMILES representation. This contradicts the preconception of the field that using the SMILES representation with an evolutionary algorithm is relatively inefficient.[26,28,29,31,33] For the global optimization of molecular properties, MolFinder employs the CSA algorithm, which has been successfully applied to many global optimization problems in various disciplines.[35–39] Compared to conventional GA, the CSA algorithm has sophisticated selection procedures to control the diversity of populations/solutions during sampling. By considering the diversity of sampled molecules, MolFinder finds novel molecules with better properties than those generated by ML-based

methods. Additionally, it is demonstrated that MolFinder successfully explores a wider range of chemical space than the other ML-based methods tested here.

Methods

Global property optimization using conformational space annealing

The goal of this study is to develop an efficient algorithm that performs global optimization of molecular properties on chemical space. We call our method MolFinder. In this study, the CSA algorithm, a highly efficient global optimization algorithm, was utilized for the global search on chemical space.[35,37,39–41] CSA combines the strengths of GA, simulated annealing,[42] and Monte-Carlo minimization.[43] It performs an extensive search during the initial stage of search and intensive optimization near many different local minima during the later stage of the search by controlling distance constraints between candidate solutions. The detailed description of the general CSA algorithm and its efficiency are discussed in detail elsewhere.[36]

MolFinder performs a global search on chemical space using the SMILES representation. The workflow of MolFinder is illustrated in Figure S1. During the search, MolFinder uses a set of molecules called a *bank*, and its size, N_{bank}, is kept constant during the search. In this study, N_{bank} is set to 1000. MolFinder starts with a predefined number of random molecules. The average distance between all pairs of molecules in the first bank is calculated, D_{avg}. The half of D_{avg} is set as an initial distance cutoff, $D_{\text{cut}} = D_{\text{avg}}/2$, which is used to keep the diversity of the bank. A distance between a pair of molecules is defined as $1 - S(m_i, m_j)$, where $S(m_i, m_j)$ is the similarity between the two molecules, m_i and m_j. In this study, a similarity between the two molecules is calculated by using the Tanimoto coefficient of their RDKit fingerprint vectors.[34]

Among N_{bank} molecules, a subset of best molecules in terms of a given objective function with a size of N_{seed} is selected as seed molecules for generating new molecules. In this study,

we set $N_{\mathrm{seed}} = 600$. Afterward, one molecule is randomly selected from this seed set, and the other from the entire bank. New molecules, child solutions, are generated from this pair through cross-over and mutation operations (Figure 1). From a single seed molecule, 40 molecules are generated by crossover. Mutation operations consist of addition, deletion, and substitution of an atom, and 20 molecules are generated by each operation, respectively. In summary, a total of 100 molecules are generated from one seed molecule.

The generated molecules are followed by local optimization. For local optimization, atoms in a molecule are randomly substituted with other elements for a certain number of times. If the objective value of a molecule becomes better, the change is accepted, otherwise rejected. In this study, we tested the two versions of MolFinder, with and without this local optimization step. Sampling with local optimization is called MolFinder-local in this paper.

The generated new molecules are used to update the bank by considering both the diversity of molecules and their objective values. First, if a new molecule has a worse objective value than the worst of the bank, it is discarded. If it is not discarded, the molecule is compared with all molecules in the bank and its nearest neighbor is identified. Then, if the distance between the molecule and its nearest neighbor is less than D_{cut}, the two molecules are considered to be in the same basin on chemical space. Thus, only one molecule with a better objective value remains. If the distance between the new molecules and its nearest neighbor in the bank is larger than D_{cut}, the new molecule is considered to represent a favorable novel region and it replaces the molecule with the worst objective value in the bank. The D_{cut} value decreases by a power of 0.98 after every generation until it reaches $D_{\mathrm{avg}}/5$. After D_{cut} becomes $D_{\mathrm{cut}}/5$, it remains constant. By using this update procedure, the CSA algorithm enables an extensive search on chemical space and prevents the premature convergence of the search.

Crossover operation

The key components of MolFinder are crossover and mutation operations using SMILES strings to generate novel molecules (Figure 1). Previously, it has been considered that performing GA with SMILES is inefficient because the random crossover and the mutation operations of SMILES strings mostly result in invalid SMILES strings.[28,29] To alleviate this invalid SMILES problem, we devised sophisticated crossover and mutation procedures to increase the success rate of valid SMILES generation.

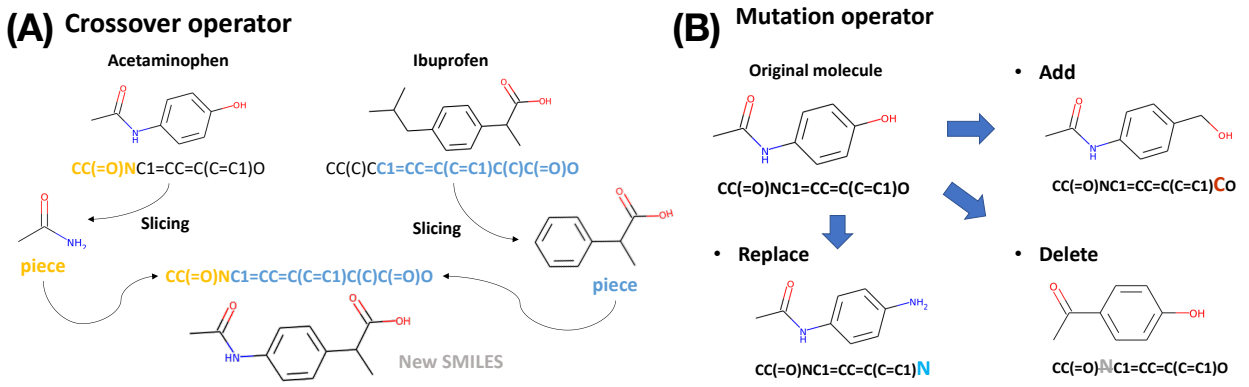

Figure 1: The crossover (A) and mutation operations (B) using SMILES strings.

The pseudocode of the crossover operation is presented in Algorithm 1. A pair of SMILES strings are truncated from both the left and the right to enhance the diversity of substructures. In other words, one string is truncated from the left and the other from the right. The positions to be truncated are selected almost randomly for both strings by considering ring structures. To generate more valid SMILES strings, truncation of a SMILES string in the middle of a ring structure is avoided. The two truncated strings are concatenated and the numbers of open and closing parentheses are counted. If they do not match, excess parentheses are removed or deficient parentheses are inserted at random positions. After fixing imbalanced parentheses, the validity of the resulting string is checked. If the concatenated string is not valid, the procedure is repeated until it results in a valid SMILES string. If a valid SMILES is not found after 30 iterations, the pair is dismissed.

Algorithm 1 Crossover operation

best SMILES $\rightarrow m_1$
random SMILES $\rightarrow m_2$
while m_{new} is invalid **do**
    Select SMILES random positions $\leftarrow m_1$, m_2
    Truncated from left SMILES $\rightarrow m_l$
    Truncated from right SMILES $\rightarrow m_r$
    $m_{new} \leftarrow$ **concatenate** m_1 and m_2 Fix imbalanced parentheses of m_{new}
    Check the validity of m_{new}
end while
Append m_{new} to the offspring list

Mutation operation

Mutation operations consist of the insertion, deletion, and substitution of atoms of a molecule. For insertion and deletion operations, an atom is inserted or deleted at a random position of a SMILES string. If the resulting string is not valid, the operation is repeated until a valid molecule is generated up to 30 times. The pseudocode of the substitution operation is shown in Algorithm 2. A random atom of a molecule is substituted with another atom considering its neighboring environment, such as the number of valences. To consider the valence of an atom properly, a SMILES string is converted to a Mol type instance of RDKit.

Algorithm 2 Substitution operation

Atom list = [B, C, N, O, F, P, S, Cl, Br, I]
Aromatic atom list = [C, N, P, O, S]
Convert a Mol-type instance of RDKit into a SMILES string
Randomly select an atom of a molecule
if Selected an atom is an aromatic atom **then**
    Replace an atom from the aromatic atom list considering valence
else
    Replace an atom from the atom list considering valence
end if
return Convert a SMILES string into a Mol-type instance of RDKit

Dataset

In this study, initial molecules were randomly sampled from the ZINC15 database,[22] which consists of purchasable drug-like molecules. As of Nov. 2019, there were over 980 million SMILES strings in ZINC15 and they were grouped as tranches based on molecular weight and logP values. We randomly sampled 1/1000 of each tranche, resulting in 982,518 SMILES strings. This subset was used as a seed set for both MolFinder and the training set for other deep-learning-based generation models.

Comparison with reinforcement-learning-based methods

To assess the efficiency of MolFinder, we compared the objective values and the diversity of generated molecules with two generative-model-based molecular property optimization approaches, ReLeaSE[11] and MolDQN.[12] ReLeaSE uses the reinforcement-learning approach[16] and a stacked-RNN model[15] to generate novel SMILES strings with desired properties. To compare with MolFinder, we used the ReLeaSE code downloaded from its Github repository.[11] The initial training of a stacked-RNN machine to learn the syntax of SMILES was performed with the training set, the random subset of ZINC15. A learning rate of 0.00005 was used. After initial training, reinforcement-learning was performed for 3000 steps to optimize the machine to produce more molecules with desired properties.

MolDQN[12] is a molecular property optimization approach based on the deep-Q-network (DQN) reinforcement learning algorithm.[16] With the MolDQN approach, a seed molecule is modified by atom addition, bond addition/deletion operations to optimize target properties. The advantage of MolDQN is that it generates valid molecules mostly because it generates a new molecule by modifying a seed molecule with the predefined operations. We downloaded the MolDQN code from its Github repository and reinforcement-learning was performed for 40,000 episodes. One episode means the completion of modifying a seed molecule. Similar to ReLeaSE, MolDQN also requires the initial training of its generative model to learn the syntax of SMILES. The generator of MolDQN was trained with the identical training set

with ReLeaSE. MolDQN simulations were performed from the seed molecule provided in their repository.

Implementation detail

MolFinder was implemented with Python version 3.7.6. To compute molecular similarities and properties, RDKit version 2019.09.3.0[34] was used. MolDQN was performed with Tensorflow version 1.15[44] and ReLeaSE with PyTorch version 1.4.[45]

Results and discussion

Optimization of drug-likeness

To assess the efficiency of molecular property optimization approaches, we sampled molecules by optimizing the following objective function, a modified drug-likeness score, S_{mQED}:

$$S_{\mathrm{mQED}}(m) = wS_{\mathrm{QED}}(m) + (1-w)S_{\mathrm{SAS}}(m) \tag{1}$$

, where $S_{\mathrm{QED}}(m)$ is the original quantitative estimate of drug-likeness (QED) score[46] of a molecule m, $S_{\mathrm{SAS}}(m)$ is the synthetic accessibility[47] of m, and w is the weight of S_{QED}. In this study, we used $w = 0.994$. The QED score ranges from 0 to 1, and more drug-like molecules have values closer to 1. The synthetic accessibility score spans from 0 to 10, and a higher score indicates that a molecule is expected to be harder to synthesize.[47] Thus a high modified QED value, S_{mQED}, indicates that a molecule has similar molecular properties to known drugs and is easy to synthesize.

To assess the optimization efficiency of ReLeaSE and MolDQN, we generated 10,000 SMILES strings with each method using S_{mQED} (eq. 1). The validity of the strings was checked and only valid ones were kept for further analysis. All SMILES strings generated by MolDQN were valid. However, after removing redundancy, only 4273 molecules remained.

This shows that more than half of the generated molecules by MolDQN were redundant. ReLeaSE generated 9821 valid SMILES strings from 10000 trials. After removing redundancy, only 1340 molecules remained. In other words, more than 80% of the generated molecules by ReLeaSE were redundant suggesting that generative models may have limitations in sampling diverse molecules. For a fair comparison, the top-1000 molecules in terms of S_{mQED} were selected from each generated set.

A comparison of the top-1000 molecules obtained with MolFinder and the other approaches demonstrates that MolFinder discovers better molecules than the other methods (Table 1 and Figure 2). MolFinder-local achieved the highest mean S_{mQED} of 1000 molecules, 0.9240. The molecule with the highest S_{mQED}, 0.9326, was also obtained with MolFinder-local. It is noticeable that the minimum S_{mQED} values obtained with both MolFinder models, 0.921 and 0.920, are significantly higher than those of the ReLeaSE and MolDQN results, which are 0.847 and 0.868, respectively. These numbers indicate that even the worst molecules generated by the MolFinder are comparable to those generated by the RL-based methods. When the two versions of MolFinder methods are compared, it is identified that MolFinder-local finds slightly better molecules than MolFinder.

Table 1: A comparison of modified drug-likeness optimization results by the MolFinder, ReLeaSE and MolDQN methods.

	ZINC	MolFinder	MolFinder-local	ReLeaSE	MolDQN
mean	0.7086	0.9237	**0.9240**	0.8473	0.8677
std	0.1248	0.0020	0.0027	0.0380	0.0240
min	0.3263	**0.9209**	0.9199	0.7570	0.8281
max	0.9224	0.9316	**0.9326**	0.9317	0.9235

Overall, the ReLeaSE results have the lowest mean and minimum S_{mQED} values. However, it found one molecule that has a higher S_{mQED} value than the best of MolFinder, but lower than that of MolFinder-local. This indicates that the molecules generated by ReLeaSE have a wide distribution in terms of S_{mQED}. Similarly, MolDQN generated a few molecules with S_{mQED} values higher than 0.9. However, the S_{mQED} values of most molecules generated by

MolDQN were distributed between 0.85 to 0.90, which were significantly lower than the MolFinder and MolFinder-local results (Figure 2).

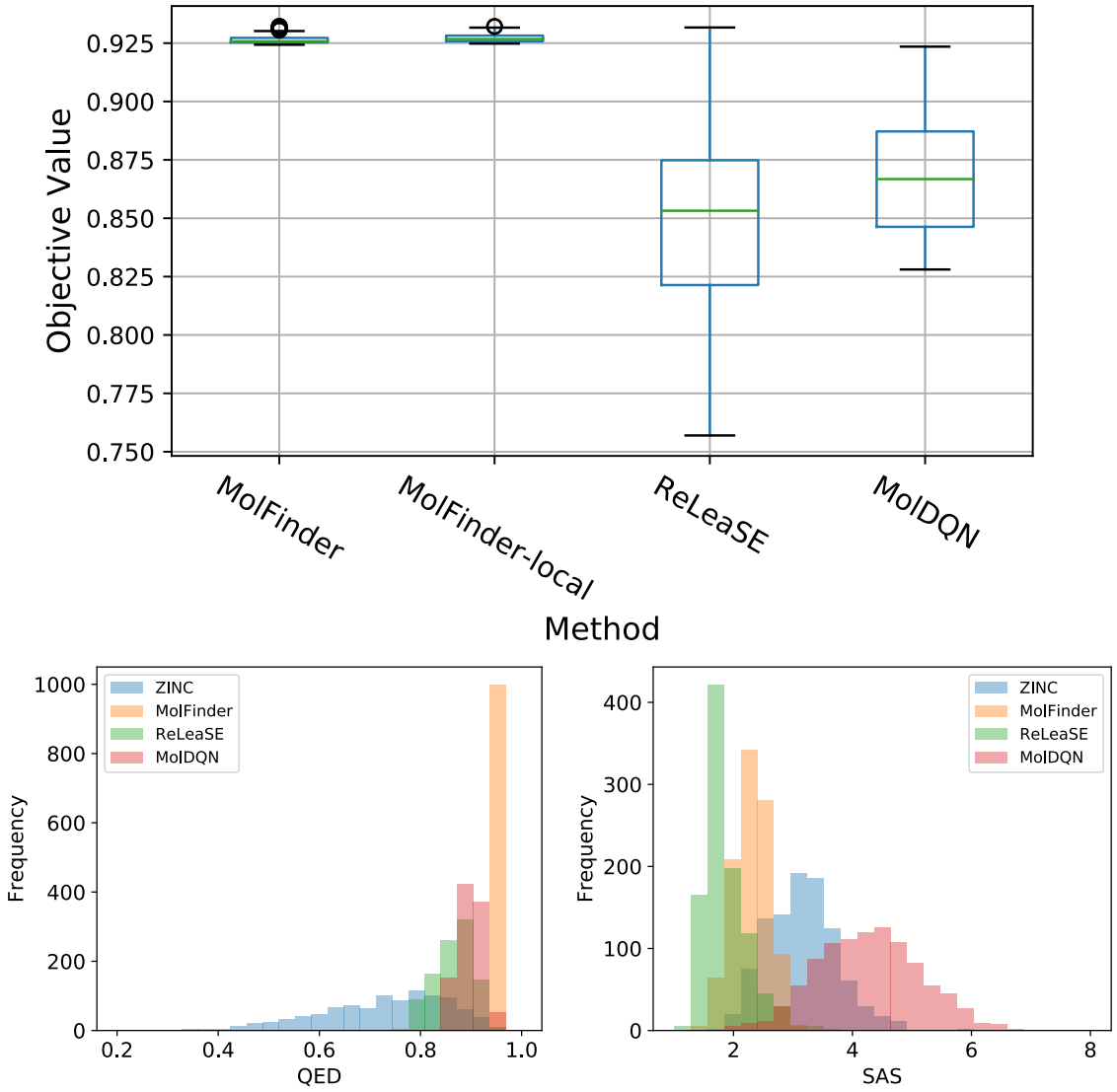

Figure 2: The boxplot of the modified drug-likeness scores of generated molecules by MolFinder, MolFinder-local, ReLeaSE, and MolDQN (top). The histogram of QED (left bottom) and SAS (right bottom) values of the generated molecules by MolFinder (orange), ReLeaSE (green), and MolDQN (red), and those of the initial ZINC15 database (blue).

For further analysis, we compared the distributions of the original QED score and the SA score independently (the bottom plots of Figure 2). The analysis shows that MolFinder results have significantly higher original QED values than the other methods (left bottom of Figure 2). All molecules generated by MolFinder had S_{QED} values of higher than 0.92.

On the other hand, the results of the other methods have lower S_{QED} values. Following MolFinder, the most frequently observed S_{QED} values of MolDQN and ReLeaSE results are centered around *0.90*. On average, MolDQN results have slightly higher S_{QED} values than the ReLeaSE results. All optimization results have higher S_{QED} values than ZINC15 on average.

In terms of synthetic accessibility, the ReLeaSE results have the lowest average S_{SAS} value meaning that they are relatively easier to synthesize, followed by the MolFinder and MolDQN results (right bottom of Figure 2). It is noticeable that the MolDQN results have significantly higher S_{SAS} values than the initial molecules from ZINC15. This suggests that MolDQN tends to optimize seed molecules by modifying them into complicated and harder ones to synthesize (Figure S4). On the other hand, the reinforced ReLeaSE is inclined to generate rather simpler molecules (Figure S5). In summary, although both ReLeaSE and MolDQN are based on the reinforcement learning algorithms, they optimize molecules in the opposite way: making molecules simpler and more complex. The S_{SAS} values of MolFinder results are distributed between those of the ReLeaSE and MolDQN results, which are also improved than the ZINC15 set (Figure S6).

The top-12 molecules discovered by MolFinder are presented in Figure 3. It appears that all molecules consist of relatively simple fragments and high S_{QED} values. All top-12 molecules in Figure 3 have low S_{SAS} values, less than 2.5, suggesting that they are readily synthesizable. It is noticeable that, even though we optimized S_{mQED} in this study, the S_{QED} values of the top-12 molecules are identical or comparable to the best reported values obtained from the sole optimization of S_{QED}.[25] In conclusion, the above results indicate that molecule optimization of S_{mQED} using MolFinder successfully generated a set of molecules with good drug-likeness and synthetic accessibility simultaneously. This clearly demonstrates that MolFinder can help accelerate the drug discovery process by generating novel drug candidates that are readily synthesizable.

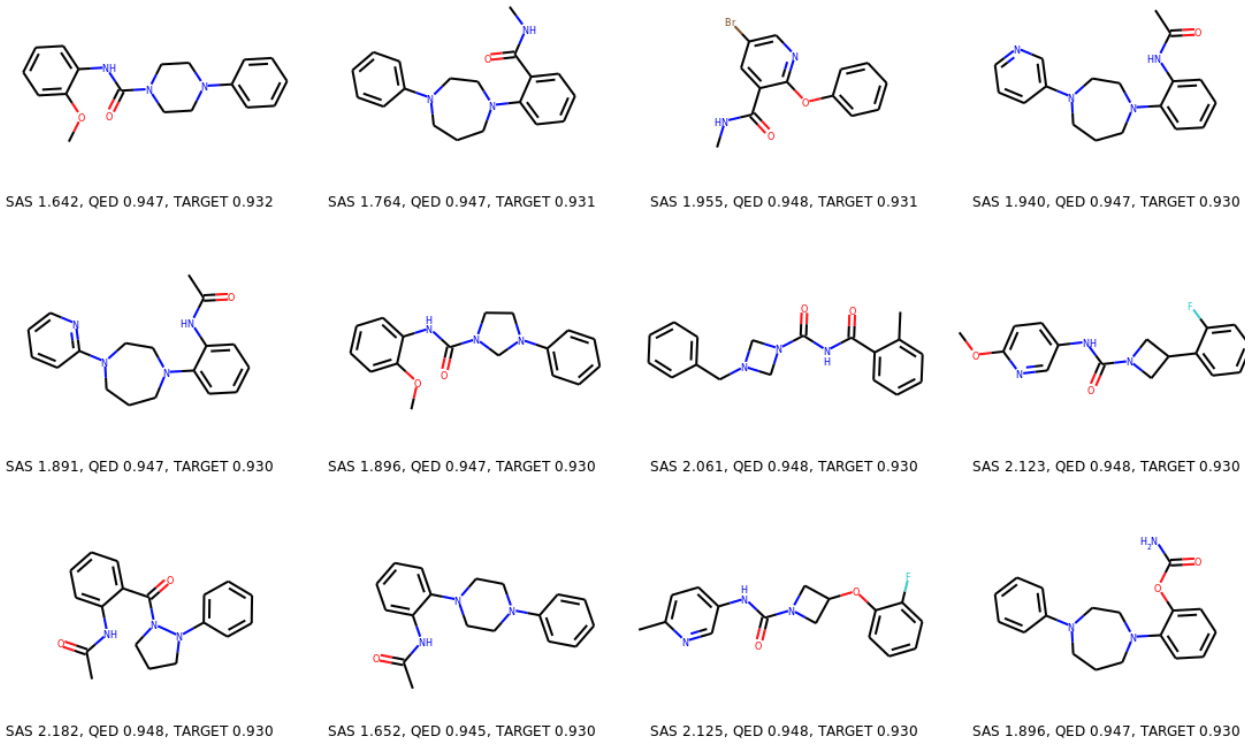

Figure 3: Top-12 molecules discovered by MolFinder. The modified drug-likeness scores (TARGET, eq. 1) and their drug-likeness (QED) and synthetic accessibility score (SAS) are presented.

Diversity of generated molecules

To assess the sampling efficiency of the tested approaches, pairwise similarities between the generated molecules were investigated (Table 2). It is demonstrated that MolFinder and MolFinder-local find more diverse sets of molecules than the other RL-based approaches. This suggests that MolFinder performs a more extensive exploration of chemical space than the others. The average pairwise similarities of molecules sampled by MolFinder and MolFinder-local were 0.3106 and 0.3211, while those of ReLeaSE and MolDQN were 0.4330 and 0.4097, respectively. From the histogram of pairwise similarities, it is evident that most pairs of molecules have similarity values between 0.1 and 0.4 (Figure S8). Although the ReLeaSE results show a peak of around 0.2, which is similar to the MolFinder results, they also include many pairs of molecules whose similarities are over 0.4. The MolDQN results have a peak of around 0.38, which is significantly larger than those of the other methods.

In other words, the molecules generated by the MolFinder methods are highly diverse while those generated by ReLeaSE and MolDQN are much more similar to each other. This implies that RL-based methods are likely to be biased and their results may be confined to a certain region of chemical space possibly due to training data dependency.

Table 2: A comparison of pairwise similarities between generated molecules by the MolFinder, ReLeaSE and MolDQN methods.

	MolFinder	MolFinder-local	ReLeaSE	MolDQN (default)	MolDQN (ZINC)
mean	**0.3106**	0.3211	0.4330	0.4097	0.3693
std	0.0716	0.0560	0.1116	0.0782	0.0719

To identify the training/initial data dependency of the methods, the distributions of generated molecules are displayed by using the t-SNE dimension reduction method.[48] A molecular similarity was calculated using the MACCS key.[49] From the plot, it is clear that MolFinder and MolFinder-local sampled different regions of chemical space compared to the initial data from ZINC15. On the t-SNE plot, MolFinder results form several distinct clusters that are widely spread over chemical space. On the other hand, molecules generated by ReLeaSE are mostly clustered at the right top of the plot, which suggests that they are similar to each other and the sampling of ReLeaSE may be biased. Also, molecules from Zinc15 are highly populated at the right top region and they are largely overlapped with the ReLeaSE results. MolDQN results overlap with the training data most. Molecules from ZINC15 and MolDQN are mostly clustered around the center and the left-center region of the plot. This indicates that molecules generated by MolDQN are highly similar to seed molecules, which may limit the sampling efficiency of the method. In summary, MolFinder and MolFinder-local explore wider regions of chemical space than the other methods.

Assessment of novelty of molecules

To further analyze the sampling efficiencies of the molecular optimization methods, the uniqueness and novelty of molecules and their Bemis–Murcko scaffolds[50] were investigated

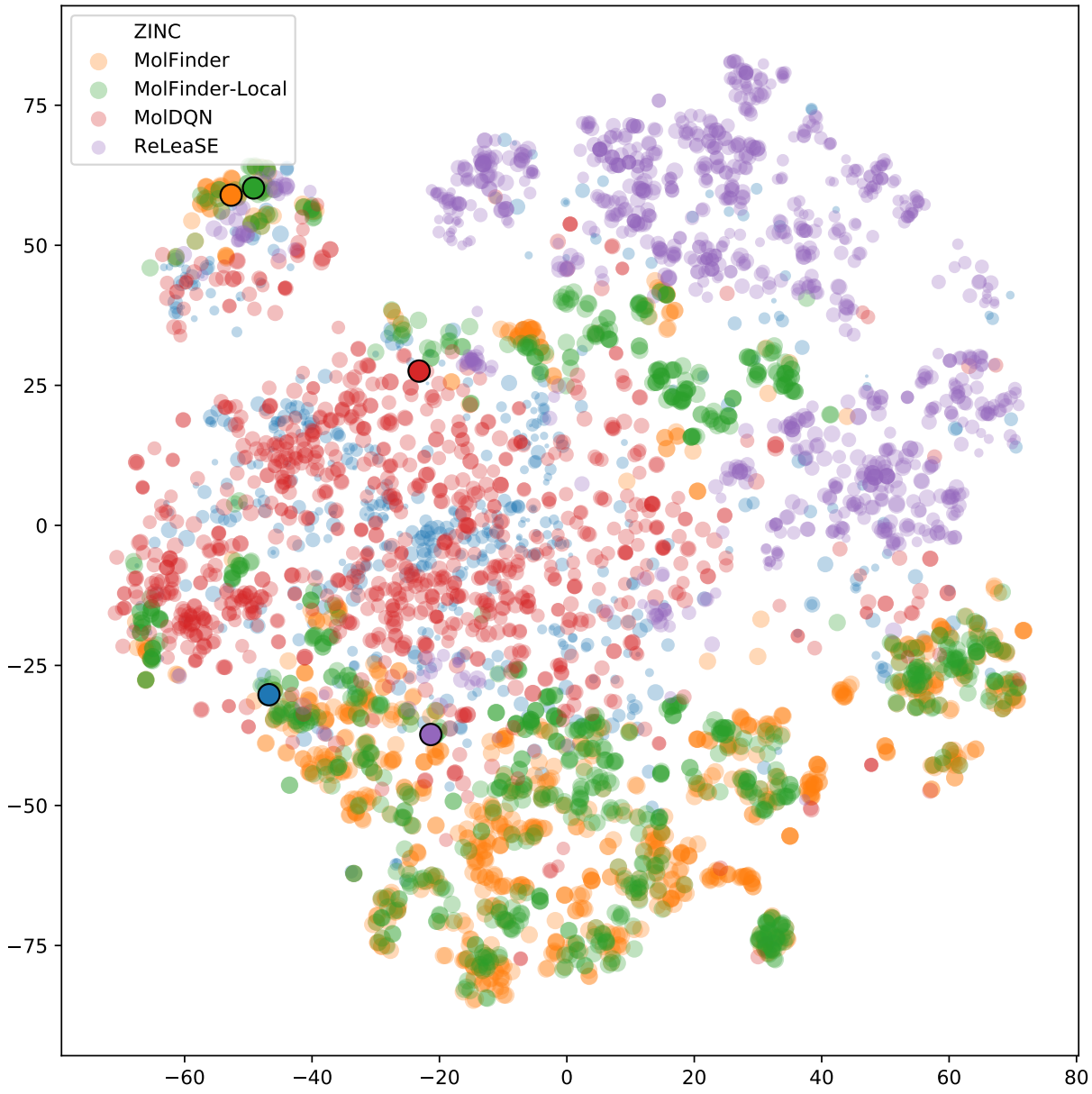

Figure 4: The t-SNE plot of the top-1000 molecules generated by MolFinder (yellow), MolFinder-local (green), MolDQN (red), and ReLeaSE (purple). For comparison, initial/seed molecules from ZINC15 (blue) are illustrated together. The sizes of circles are proportional to the molecules' S_{mQED} values. The best molecule generated by each method is emphasized with black border lines.

(Table 3). Almost all molecules generated by MolFinder and MolFinder-local were novel, absent in the input database. Only one molecule generated by MolFinder was found in the input database and none by MolFinder-local. Thirty-three molecules generated by ReLeaSE

were redundant. In terms of scaffolds, MolDQN found the most unique scaffolds, 880. However, as identified by higher SA scores in Figure 2, MolDQN results have relatively complex chemical structures, such as many fused rings, which make them hard to synthesize and less practical (Figure S4). The MolFinder and MolFinder-local methods generated 860 and 828 scaffolds, respectively, which are comparable to the MolDQN results. However, their S_{SAS} values are significantly lower than those of the MolDQN results (Figure 2). In other words, most molecules discovered by MolFinder were drug-like and reasonably easy enough to synthesize (Figure S6 and S7). It is noticeable that ReLeaSE generated only 213 unique scaffolds, which are remarkably smaller than the other methods. Many molecules generated by ReLeaSE were identified to have similar scaffolds and only peripheral groups were different (Figure S5). This suggests that the reinforced generator of ReLeaSE may be biased to yield only similar molecules, which limit the efficiency of RL-based models.

Table 3: A comparison of uniqueness and novelty of generated molecules and their scaffolds

Method	Unique	Novel (M)	Scaffolds (N)	$P_{\mathrm{scaffolds}}$ (N/M)	Novel scaffold %
ZINC	1000	-	956	0.956	-
MolFinder	1000	1000	860	0.860	**99.2**
MolFinder-local	1000	**1000**	828	0.828	98.6
MolDQN	1000	997	**880**	**0.883**	96.1
ReLeaSE	967	967	213	0.220	92.0

Additionally, the percentages of novel scaffolds were investigated. If a scaffold was not found in the initial dataset, it was considered novel. MolFinder results showed the highest percentage of a novel scaffold, 99.2%. The percentages of novel scaffolds of the ReLeaSE and MolDQN results, 96.1% and 92.0%, were lower than those of MolFinder and MolFinder-local. This demonstrates that the MolFinder methods not only optimize a target property more efficiently but also perform a wider exploration of chemical space than the other methods.

Generating similar molecules to a reference

Designing novel molecules based on a specific scaffold or a core structure is a commonly used approach for molecular design. Thus, generating molecules with desired properties while preserving a specific scaffold has practical advantages. To benchmark this, we optimized the following objective function used by Zhou et al.[12] using MolFinder and MolDQN:

$$f(m) = wS_{\mathrm{sim}}(m; m_{\mathrm{ref}}) + (1 - w)S_{\mathrm{QED}}(m) \tag{2}$$

where $S_{\mathrm{sim}}(m, m_{\mathrm{ref}})$ is the Tanimoto similarity between a molecule m and a reference molecule m_{ref} calculated with the Morgan fingerprint and w is the weight coefficient of the similarity term. Here, we set $w = 0.8$.

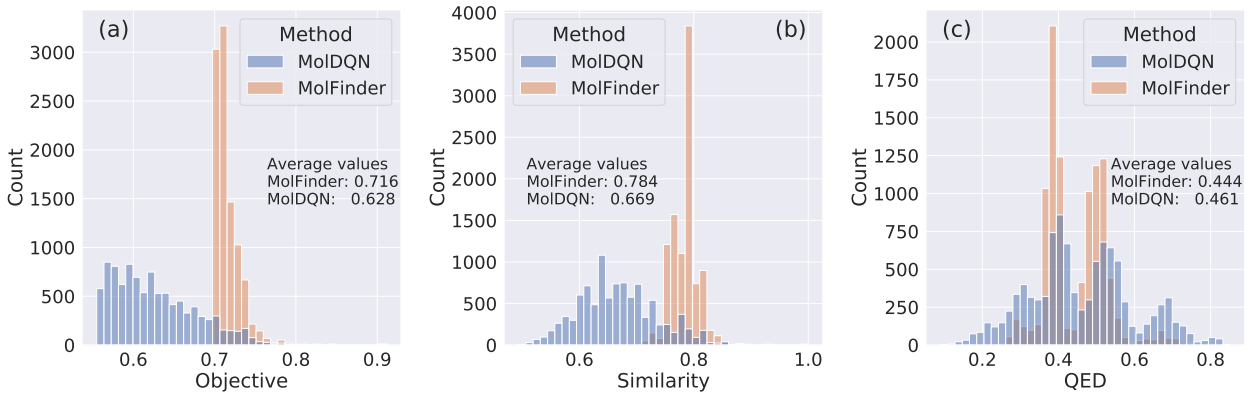

Figure 5: Histograms of (a) objective values (eq. 2), (b) similarities to the reference molecules, and (c) drug-likeness scores (QED) of molecules generated by MolFinder (orange) and MolDQN (blue).

Independent molecular generation calculations were repeated ten times using MolFinder and MolDQN based on the same reference molecule used to benchmark MolDQN (PubChem CID: 174590).[12] Each MolDQN simulation was performed for 40,000 episodes and only the best 1000 non-redundant molecules in terms of the objective value (eq. 2) were analyzed. Thus, 10,000 molecules were generated by MolFinder and MolDQN, respectively, and they are analyzed here.

It is demonstrated that the molecules generated by MolFinder have remarkably higher

objective values and similarities than those generated by MolDQN (Figure 5). The average objective value of the MolFinder results was 0.716, while that of the MolDQN results was 0.628. All molecules generated by MolFinder have higher objective function values over 0.7, while MolDQN results peaked around 0.6.

This difference is mainly attributed to the higher similarity to the reference molecule ($S_{\mathrm{sim}}(m; m_{\mathrm{ref}})$ in eq. 2). The molecules generated by MolFinder had an average similarity of 0.784 to the reference. However, the molecules generated by MolDQN were less similar to the reference with an average similarity of 0.669. This result shows that MolFinder results are much similar to the reference as intended. In terms of the QED, the MolDQN results were slightly better than the MolFinder results, 0.461 to 0.444, while the difference is much smaller than that of the similarity. It is not clear whether such a small difference in QED, 0.017, will lead to a significant difference in the final quality of generated molecules. In summary, these results suggest that MolFinder outperforms MolDQN in generating molecules that have desired properties and are similar to a given reference molecule, simultaneously.

Conclusion

In this study, we presented a new molecule optimization approach, MolFinder, based on the efficient global optimization of molecular properties using the SMILES representation. This method performs a global search on chemical space by using the crossover and mutation operations of the SMILES representation, which makes the method computationally efficient and straightforward to implement. Our work indicates that applying evolutionary algorithms based on the SMILES representation to molecular property optimization is promising, which has been overlooked by the field despite its simplicity. We showed that MolFinder finds better molecules than the ML-based molecular property optimization methods in terms of a given objective function. In addition, it is also demonstrated that MolFinder samples a more diverse set of molecules than the other tested methods.

The key components of the efficiency of MolFinder are the following two. First, MolFinder uses the sophisticated crossover and mutation operations of SMILES to increase the success rate of the operations. Second, the diversity of the bank of molecules was kept during the exploration of chemical space as much as possible, which is one of the key aspects of the CSA algorithm. One common limitation of conventional GA is that all solutions become highly similar to each other, meaning that the sampling is trapped in a local minimum or a set of local minima. In many previous studies using CSA, it has been shown that keeping the diversity of the bank high is critical in efficient search on multi-dimensional hyper-spaces.[36–39] However, despite the efficiency of MolFinder, we cannot completely rule out the possibility of any sampling bias caused by crossover and mutation operations.

The results presented in this paper clearly demonstrate that applying an evolutionary algorithm with the SMILES representation can be an effective strategy for molecular optimization, which is contrary to the conventional notion.[26,28,29] Thus our results will facilitate the development of new computational molecular design approaches based on the SMILES representation, which is advantageous in terms of its interpretability, manipulation and sharing data with other researches. In conclusion, we believe that MolFinder is an alternative complementary approach to existing GA-based as well as ML-based methods and paves a new path for the inverse design of molecules via property optimization.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grants funded by the Korean government (MSIT) (Nos.2019M3E5D4066898).

Supporting Information Available

References

(1) Kuhn, C.; Beratan, D. N. Inverse strategies for molecular design. *Journal of Physical Chemistry* **1996**, *100*, 10595–10599.

(2) Sanchez-Lengeling, B.; Aspuru-Guzik, A. Inverse molecular design using machine learning: Generative models for matter engineering. *Science* **2018**, *361*, 360–365.

(3) Schneider, P. et al. Rethinking drug design in the artificial intelligence era. *Nature Reviews Drug Discovery* **2020**, *19*, 353–364.

(4) Elton, D. C.; Boukouvalas, Z.; Fuge, M. D.; Chung, P. W. Deep learning for molecular design - A review of the state of the art. *Molecular Systems Design and Engineering* **2019**, *4*, 828–849.

(5) Weininger, D. SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules. *Journal of Chemical Information and Computer Sciences* **1988**, *28*, 31–36.

(6) Kingma, D. P.; Welling, M. Auto-encoding variational bayes. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings* **2014**, 1–14.

(7) Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Central Science* **2018**, *4*, 268–276.

(8) Lim, J.; Ryu, S.; Kim, J. W.; Kim, W. Y. Molecular generative model based on con-

ditional variational autoencoder for de novo molecular design. *Journal of Cheminformatics* **2018**, *10*, 1–9.

(9) Yuan, W.; Jiang, D.; Nambiar, D. K.; Liew, L. P.; Hay, M. P.; Bloomstein, J.; Lu, P.; Turner, B.; Le, Q.-T.; Tibshirani, R., et al. Chemical space mimicry for drug discovery. *Journal of chemical information and modeling* **2017**, *57*, 875–882.

(10) Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H. Molecular de-novo design through deep reinforcement learning. *Journal of Cheminformatics* **2017**, *9*, 1–14.

(11) Popova, M.; Isayev, O.; Tropsha, A. Deep reinforcement learning for de novo drug design. *Science Advances* **2018**, *4*, 1–15.

(12) Zhou, Z.; Kearnes, S.; Li, L.; Zare, R. N.; Riley, P. Optimization of Molecules via Deep Reinforcement Learning. *Scientific Reports* **2019**, *9*, 1–10.

(13) Zhavoronkov, A. et al. Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nature Biotechnology* **2019**, *37*, 1038–1040.

(14) Putin, E.; Asadulaev, A.; Ivanenkov, Y.; Aladinskiy, V.; Sanchez-Lengeling, B.; Aspuru-Guzik, A.; Zhavoronkov, A. Reinforced Adversarial Neural Computer for de Novo Molecular Design. *Journal of Chemical Information and Modeling* **2018**, *58*, 1194–1204.

(15) Joulin, A.; Mikolov, T. Inferring algorithmic patterns with stack-augmented recurrent nets. *Advances in Neural Information Processing Systems* **2015**, *2015-January*, 190–198.

(16) Mnih, V. et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533.

(17) Kearnes, S.; Li, L.; Riley, P. Decoding Molecular Graph Embeddings with Reinforcement Learning. *arXiv:1904.08915* **2019**,

(18) Kwon, Y.; Yoo, J.; Choi, Y. S.; Son, W. J.; Lee, D.; Kang, S. Efficient learning of non-autoregressive graph variational autoencoders for molecular graph generation. *Journal of Cheminformatics* **2019**, *11*, 1–10.

(19) Chen, G.; Shen, Z.; Iyer, A.; Ghumman, U. F.; Tang, S.; Bi, J.; Chen, W.; Li, Y. Machine-learning-assisted de novo design of organic molecules and polymers: Opportunities and challenges. *Polymers* **2020**, *12*.

(20) Schwalbe-Koda, D.; Gómez-Bombarelli, R. Generative Models for Automatic Chemical Design. *Lecture Notes in Physics* **2020**, *968*, 445–467.

(21) Gantzer, P.; Creton, B.; Nieto-Draghi, C. Inverse-QSPR for de novo Design: A Review. *Molecular Informatics* **2020**, *39*, 1–21.

(22) Sterling, T.; Irwin, J. J. ZINC 15 – Ligand Discovery for Everyone. *Journal of Chemical Information and Modeling* **2015**, *55*, 2324–2337, PMID: 26479676.

(23) Gaulton, A.; Bellis, L. J.; Bento, A. P.; Chambers, J.; Davies, M.; Hersey, A.; Light, Y.; McGlinchey, S.; Michalovich, D.; Al-Lazikani, B.; Overington, J. P. ChEMBL: A large-scale bioactivity database for drug discovery. *Nucleic Acids Research* **2012**, *40*, 1100–1107.

(24) Kim, S.; Chen, J.; Cheng, T.; Gindulyte, A.; He, J.; He, S.; Li, Q.; Shoemaker, B. A.; Thiessen, P. A.; Yu, B.; Zaslavsky, L.; Zhang, J.; Bolton, E. E. PubChem 2019 update: Improved access to chemical data. *Nucleic Acids Research* **2019**, *47*, D1102–D1109.

(25) Leguy, J.; Cauchy, T.; Glavatskikh, M.; Duval, B.; Mota, B. D. EvoMol: a flexible and interpretable evolutionary algorithm for unbiased de novo molecular generation. *Journal of Cheminformatics* **2020**, 1–19.

(26) Henault, E. S.; Rasmussen, M. H.; Jensen, J. H. Chemical Space Exploration: How Genetic Algorithms Find the Needle in the Haystack. *PeerJ* **2020**, 1–15.

(27) Nigam, A.; Friederich, P.; Krenn, M.; Aspuru-Guzik, A. Augmenting genetic algorithms with deep neural networks for exploring the chemical space. *arXiv preprint arXiv:1909.11655* **2019**,

(28) Brown, N.; Fiscato, M.; Segler, M. H.; Vaucher, A. C. GuacaMol: Benchmarking Models for de Novo Molecular Design. *Journal of Chemical Information and Modeling* **2019**, *59*, 1096–1108.

(29) Jensen, J. H. A graph-based genetic algorithm and generative model/Monte Carlo tree search for the exploration of chemical space. *Chemical Science* **2019**, *10*, 3567–3572.

(30) Yoshikawa, N.; Terayama, K.; Sumita, M.; Homma, T.; Oono, K.; Tsuda, K. Population-based De Novo Molecule Generation, Using Grammatical Evolution. *Chemistry Letters* **2018**, *47*, 1431–1434.

(31) Devi, R. V.; Sathya, S. S.; Coumar, M. S. Evolutionary algorithms for de novo drug design - A survey. *Applied Soft Computing Journal* **2015**, *27*, 543–552.

(32) Virshup, A. M.; Contreras-García, J.; Wipf, P.; Yang, W.; Beratan, D. N. Stochastic voyages into uncharted chemical space produce a representative library of all possible drug-like compounds. *Journal of the American Chemical Society* **2013**, *135*, 7296–7303.

(33) Hartenfeller, M.; Schneider, G. Enabling future drug discovery by de novo design. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2011**, *1*, 742–759.

(34) Landrum, G. RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling. 2013.

(35) Lee, J.; Scheraga, H. a.; Rackovsky, S. New optimization method for conformational energy calculations on polypeptides: Conformational space annealing. *Journal of Computational Chemistry* **1997**, *18*, 1222–1232.

(36) Joung, I. S.; Kim, J. Y.; Gross, S. P.; Joo, K.; Lee, J. Conformational Space Annealing explained: A general optimization algorithm, with diverse applications. *Computer Physics Communications* **2018**, *223*, 28–33.

(37) Lee, J.; Lee, I.-H.; Joung, I.; Lee, J.; Brooks, B. R. Finding multiple reaction pathways via global optimization of action. *Nature Communications* **2017**, *8*, 15443.

(38) Lee, J.; Gross, S. P.; Lee, J. Modularity optimization by conformational space annealing. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* **2012**, *85*, 056702.

(39) Lee, J.; Lee, J.; Sasaki, T. N.; Sasai, M.; Seok, C.; Lee, J. De novo protein structure prediction by dynamic fragment assembly and conformational space annealing. *Proteins: Structure, Function and Bioinformatics* **2011**, *79*, 2403–2417.

(40) Lee, J.; Lee, I.-H.; Lee, J. Unbiased global optimization of Lennard-Jones clusters for N $<$ or =201 using the conformational space annealing method. *Physical Review Letters* **2003**, *91*, 080201.

(41) Joo, K.; Lee, J.; Kim, I.; Lee, S. J.; Lee, J. Multiple sequence alignment by conformational space annealing. *Biophysical journal* **2008**, *95*, 4813–4819.

(42) Scott, K.; Gelatt, C. D.; Vecchi, M. P. Optimization by simulated annealing. *Science* **1983**, *220*, 671–680.

(43) Li, Z.; Scheraga, H. A. Monte Carlo-minimization approach to the multiple-minima problem in protein folding. *Proc. Nat. Acad. Sci. USA* **1987**, *84*, 6611–6615.

(44) Abadi, M. et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015; `http://tensorflow.org/`, Software available from tensorflow.org.

(45) Paszke, A. et al. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc., 2019; pp 8024–8035.

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

(47) Ertl, P.; Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics* **2009**, *1*, 8.

(48) van der Maaten, L.; Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research* **2008**, *9*, 2579–2605.

(49) Durant, J. L.; Leland, B. A.; Henry, D. R.; Nourse, J. G. Reoptimization of MDL keys for use in drug discovery. *Journal of Chemical Information and Computer Sciences* **2002**, *42*, 1273–1280.

(50) Bemis, G. W.; Murcko, M. A. The properties of known drugs. 1. Molecular frameworks. *Journal of Medicinal Chemistry* **1996**, *39*, 2887–2893.

Graphical TOC Entry