Berenger and Tsuda
Journal of Cheminformatics (2021) 13:88
https://doi.org/10.1186/s13321-021-00566-4

Journal of Cheminformatics

RESEARCH ARTICLE

Open Access

Molecular generation by Fast Assembly of (Deep)SMILES fragments

Francois Berenger* and Koji Tsuda

Abstract

Background: In recent years, *in silico* molecular design is regaining interest. To generate on a computer molecules with optimized properties, scoring functions can be coupled with a molecular generator to design novel molecules with a desired property profile.

Results: In this article, a simple method is described to generate only valid molecules at high frequency (> 300,000 molecule/s using a single CPU core), given a molecular training set. The proposed method generates diverse SMILES (or DeepSMILES) encoded molecules while also showing some propensity at training set distribution matching. When working with DeepSMILES, the method reaches peak performance (> 340,000 molecule/s) because it relies almost exclusively on string operations. The "Fast Assembly of SMILES Fragments" software is released as open-source at https://github.com/UnixJunkie/FASMIFRA. Experiments regarding speed, training set distribution matching, molecular diversity and benchmark against several other methods are also shown.

Keywords: Molecular generation, Molecular fragments, SMILES, DeepSMILES

Introduction

In recent years, there has been a surge of methods developed for in silico molecular generation. Mostly using deep neural networks [1–18], but not only [19–24]. Some authors use much simpler methods and the present contribution falls into this category. Notably, Polischuk [19, 25] uses molecular fragments and generates only valid molecules, while allowing some control [25] over the molecular diversity, novelty and synthetic complexity of the generated molecules. Kwon et al. [20] use direct crossover and mutation operators over SMILES strings, combined with Conformational Space Annealing [26]. Their method does not require a training set but can generate invalid SMILES. Yoshikawa et al. [27] use a population-based grammatical evolution approach (ChemGE). While their method is fast and inherently parallel, it requires an initial population of molecules and can generate invalid SMILES. Nigam et al. [21] generate molecules by Gibbs sampling of SELFIES [28]. Their approach generates only valid molecules and does not require a training set. However, it requires translating molecules to/from SELFIES [28] (a recently developed linear encoding of molecular graphs). Brown et al. [23], Jensen et al. [22] and Leguy et al. [24] use a genetic algorithm over molecular graphs. Jensen's method [22] doesn't require task-specific model training and generates only valid molecules. Leguy et al. [24] use an evolutionary algorithm sequentially building a molecular graph using seven mutation operators. Their method also does not require a training set and generates only valid molecules.

The hereby proposed method works directly at the SMILES level. It generates only valid SMILES and thus valence-correct molecules. Simplified Molecular Input Line Entry System (SMILES [29]) is a molecular file format specifying a linear encoding of molecular graphs. SMILES are a compact way to store molecules on computers. The format is supported by all chemoinformatics toolkits and hence widespread. For rather

*Correspondence: berenger@k.u-tokyo.ac.jp
Graduate School of Frontier Sciences, The University of Tokyo, 5-1-5 Kashiwa-no-ha, Kashiwa, Chiba 277-8561, Japan

© The Author(s) 2021. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

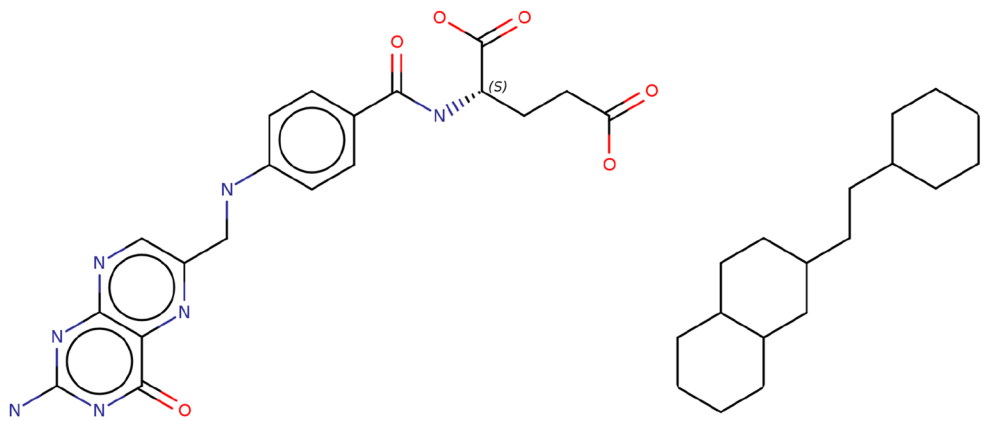

Fig. 1 Folic acid (left) and its Bemis-Murcko scaffold (right)

small molecules, SMILES are human-readable. For the remaining of this article, it is only necessary to know that SMILES are strings made of balanced parentheses (indicating possibly nested branches of a linearized tree data structure), bracket atoms (an atom between brackets carries special properties[1]), digits indicating ring opening and closures on the molecular graph plus other characters listing atoms and the bonds between them. For more details, see the Open SMILES specification [30] or the seminal paper [29]. On the other hand, DeepSMILES [31] is a recently proposed variant of SMILES, designed to ease in-silico molecular generation by making it harder to generate syntactically invalid SMILES (also one of the goals of SELFIES [21]). DeepSMILES allows two options: (i) avoiding branch opening parentheses and/or (ii) avoiding ring opening numbers. In this study, only the DeepSMILES flavor without ring opening numbers is considered.

In the experiments, to quantify molecular diversity in a dataset and the molecules generated from it, the count of unique Bemis-Murcko scaffolds [32] (Fig. 1) is monitored. In the remaining of this article, the datasets used, the method itself as well as computational experiments are presented and discussed.

Methods

The GuacaMol de Novo molecular design benchmark

GuacaMol [33–36] consists in a benchmark suite for molecular generators. The benchmark tasks measure the fidelity of models to reproduce the property distribution of a training set made of ChEMBL [37] 24 molecules[2], the ability to generate novel molecules, the exploration and exploitation of the chemical space and several optimization tasks. In this study, since no molecular optimization was performed, only the molecular generation task was used. GuacaMol allows to compare performance against a few methods with published results, using a variety of metrics. All metrics are normalized; zero being the worst score and one the best.

GuacaMol metrics: *Validity* [33]: measures if the generated molecules are realistic (e.g. SMILES is valid according to RDKit [38]). *Uniqueness* [33]: assesses whether a model generates unique molecules (i.e. few or no duplicate canonical SMILES). *Novelty* [33]: assesses whether a model generates molecules which are not present in the training set. *Fréchet ChemNet Distance* [39]: measures how close the distributions of generated molecules are to training set ones. *Kullback-Leibler (KL) divergence* [40]: measures how well a probability distribution approximates another one. For this benchmark, the probability distributions of several physicochemical descriptors [33] are compared. This metric also captures molecular diversity (given a physicochemical property distribution, the generated molecules should be as diverse as training set ones).

Some methods with published GuacaMol results [33]: *Random sampler* [33]: a baseline model only random sampling the training set. *SMILES LSTM* [3, 33]: a Long-Short-Term Memory (LSTM) neural network that predicts the next character for partial SMILES strings. *Graph MCTS* [22, 33]: Jensen's Graph-based Monte Carlo Tree Search molecular generator. *AAE*: an Adversarial AutoEncoder [41, 42]. *ORGAN*: Objective-Reinforced Generative Adversarial Network [41–43]. This deep-learning model architecture combines a generator and a discriminator network to generate molecules. *VAE*: a Variational AutoEncoder [41, 42]. This deep learning model learns a representation of molecules as latent

[1] Like explicit hydrogens, a formal charge or a specific isotope number.

[2] I.e. only molecules which have been synthesized and wet-lab tested.

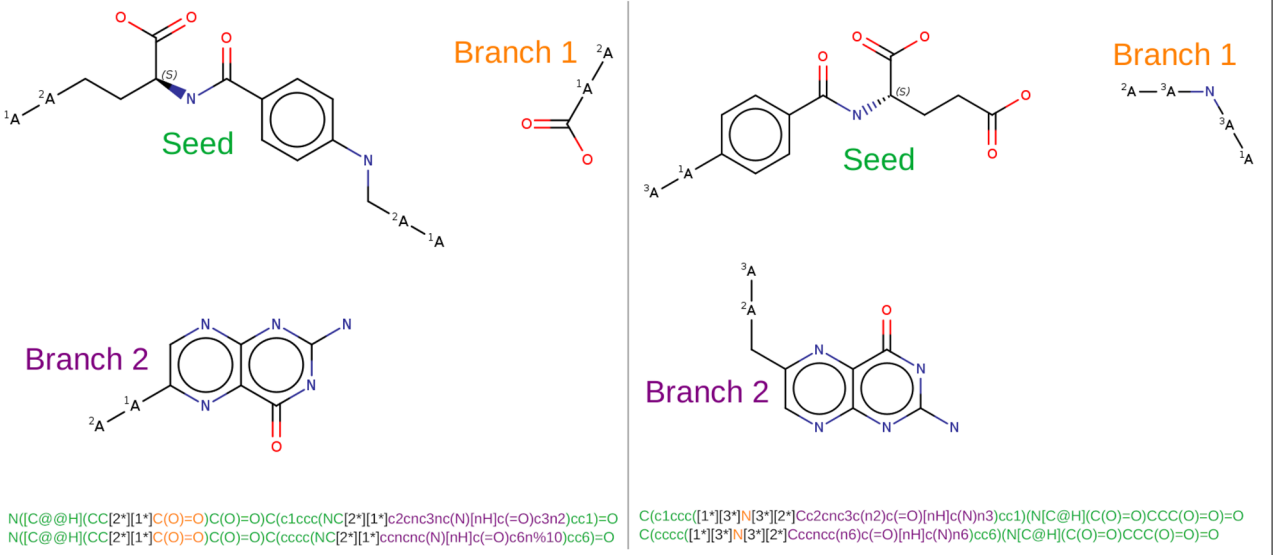

Fig. 2 Two fragmentations of folic acid (vitamin B_9). Under each mixture of fragments, the (not canonical but randomized) SMILES and corresponding DeepSMILES-without-ring-openings are shown. Tagged cleaved bonds (in black) are the wildcard atom pairs [x*][y*] with the isotope numbers x and y encoding atom types directly to the left and right of the cleaved bond. Colors indicate the seed fragment (in green), the first branch fragment (in orange) and the second branch fragment (in purple)

vectors in a continuous space. The network architecture includes an encoder network that converts SMILES strings to latent vectors, and a decoder performing the reverse operation.

Datasets

In the experiments, three datasets were used, in addition to the dataset internal to the GuacaMol benchmark. For the molecular generation speed benchmark, a random sample of one million molecules [44] from the GDB-13 [45] was used as the training set, so that results can be compared to the published results of Arús-Pous et al. [8]. For the molecular diversity and training set distribution matching experiments, two more datasets were used. To represent drug-like molecules, a bootstrap sample of 100,000 molecules was drawn from ChEMBL-28 [37]. To represent natural products, a bootstrap sample of 20,000 molecules was drawn from the Traditional Chinese Medicine Database at Taiwan [46] (this database is much smaller than ChEMBL).

The hereby proposed method is parameterized by a molecular fragmentation scheme and an atom typing scheme. Any molecular fragmentation scheme can be used, as long as it doesn't cut rings (e.g. BRICS [47] or RECAP [48]). Many atom typing schemes could be used.

Fragmenting molecules In the experiments, the following ad-hoc fragmentation scheme was used to identify then select some cleavable bonds. Only single bonds between heavy atoms not in rings can be selected. Furthermore, the bond must not be connected to a stereo center nor involved in cis-trans isomerism (an attempt at preserving the stereochemistry of fragments, if present). Cleaved bonds are chosen randomly without replacement from this list, in order to obtain n fragments. By default, a fragment molecular weight (MW) of 150Da is used and the recommended number of fragments for molecule m is given by:

$$num\_frags(m) = round\left(\frac{MW(m)}{150}\right)$$

This fragment weight parameter is just used as a hint to decide in how many fragments the given molecule must be cut (it is not strictly enforced). Since the fragmentation process is controlled by a random seed, if one requires more fragments from a given dataset, doing several passes with different seeds generates more fragments. As in Arús-Pous [9], randomized SMILES are used (instead of canonical ones) so that the same molecule does not always result in the same set of fragments (e.g. which fragment is a prefix of the SMILES, is the fragment written from left-to-right or the opposite). Duplicate fragments are not removed, because the fact that a fragment is found multiple times correlates with the natural occurring frequency of this fragment in a dataset. Also, removing duplicates might require a canonicalization step and would only be required (to reduce memory usage) if one is fragmenting a truly huge dataset.

```
C(c1ccc([1*][3*]N[3*][2*]Cc2cnc3c(n2)c(=O)[nH]c(N)n3)cc1)(N[C@H](C(O)=O)CCC(O)=O)=O
00111111222222222222222222222222233222332222223222111100111111122322211111211100O
SSSSSSSSTTTTTTTTBUUUUUUUUCCCCCCCCCCCCCCCCCCCCCCCCCCCCCSSSSSSSSSSSSSSSSSSSSSSSSSSSS
```

Fig. 3 First line: SMILES string with tagged cleaved bonds. Second line: SMILES branch nesting depth of each character from the first line (0: not in a branch; 1: inside a branch; 2: inside a branch inside a branch, etc). Third line: S characters mark the "seed fragment". T (resp. U) characters mark the first (resp. second) tagged cleaved bond. B marks the only heavy atom of the first "branch fragment". C characters mark the atoms and bonds of the second "branch fragment". The second "branch fragment" starts at SMILES branch nesting depth 2. This fragment cannot continue once the nesting depth becomes lower. The corresponding fragmented molecule can be seen on Fig. 2 right

Typing atoms In experiments and as in previous work [49], the following atom typing scheme inspired from atom pairs [50] was used: an atom a_i in molecule m is identified by the tuple

$$type(a_i) = (\pi, e, h, f)$$

where π is the number of pi electrons, e the chemical element symbol, h the number of bonded heavy atoms and f the formal charge. Stereochemistry is ignored.

Extended bond typing While the bond order (BO) is a natural bond typing scheme, to better preserve some of the structure of a molecular training set[3], it is useful to extend bond typing to make it more precise. Assuming b_j is a bond in molecule m between atoms a_i and a_{i+1}:

$$type(b_j) = (type(a_i), BO(b_j), type(a_{i+1}))$$

Note that if $type(a_i) \neq type(a_{i+1})$, the bond is directed. In fact, several published methods use some kind of extended bond typing scheme (eMolFrag [51], CReM [19, 25]).

Tagging cleaved bonds Once some bonds have been selected for cleavage, this can be encoded into a valid SMILES string where unspecified atoms (the '*' character in SMILES notation) are introduced and isotope numbers are used to encode the atom types of the atoms surrounding a cleaved bond. For example, tagging a cleaved bond in the SMILES for ethanol ('OCC') would give: 'O[2*][1*]CC'. Where isotope numbers one and two are indexes into an atom type dictionary (a mapping from integer to atom type). In the following, if a fragment is a prefix of a SMILES string, it is called a "seed fragment". All other fragments are called "branch fragments".

Fragmenting molecules with tagged cleaved bonds A SMILES string with tagged cleaved bonds can be transformed into a mixture of fragments. This is not strictly necessary, as long as the molecular fragment assembler can recognize tagged cleaved bonds. However, this is useful to visualize the fragments with a 2D molecular viewer. The SMILES 'O[2*][1*]CC' becomes 'O[2*][1*].[2*][1*]CC'. When extracting a fragment from a SMILES with tagged cleaved bonds, care must be taken to not extend the fragment past (or shorter than) its SMILES branch nesting depth. i.e. the nesting of parentheses pairs inside the SMILES string must be taken into account (Fig. 3).

Indexing molecules with tagged cleaved bonds For fast molecular generation, it is necessary to index all the fragmented molecules first. All seed fragments are stored into an array. Each branch fragment is stored into a hash-table of arrays where the prefix tagged cleaved bond is used as a key and the remaining of the fragment is appended to the array of all values for this key. For a fragment with several cleaved bonds, the first cleaved bond appearing in the fragment's SMILES (when reading it from left to right) will determine under which cleaved bond type this fragment is registered. Our method uses randomized SMILES rather than canonical ones, in order to avoid any bias that could be introduced by the canonicalization procedure.

Generating molecules In essence, some kind of Markov sampling of the previously created data structure is performed, until the generated string has no tagged cleaved bond left. The algorithm is: (i) a seed fragment is uniformly drawn from the array of seed fragments. (ii) as long as the string under construction contains tagged cleaved bonds, each of them is deleted and replaced by a uniformly drawn fragment from the array of fragments with the tagged cleaved bond as the key. When generating the right flavor of DeepSMILES, such a fragment assembly algorithm can be performed using almost exclusively string operations. However, when generating SMILES, an extra step renumbering ring opening and closure numbers is required, to avoid number collisions between rings from different fragments.

Software implementation Typing atoms and bonds, selecting and tagging cleaved bonds is done by a Python script using RDKit [38]. For performance and correctness reasons [52], indexing fragments and fragment assembly is done by an OCaml [53] program reading SMILES with tagged cleaved bonds. The program is named FASMIFRA for "Fast Assembly of SMILES Fragments".

[3] Also, to prevent molecular generation from creating new/never-seen bonds and so to favor the synthesizability of the generated molecules.

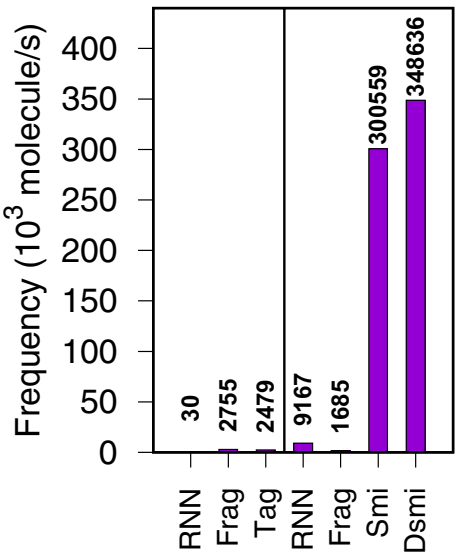

Fig. 4 Model training (left) and sampling speed (right). RNN: Recurrent Neural Network (numbers cited from literature [8]); Frag (train): molecular fragmentation (Python/RDKit); Tag: tagging cleaved bond (proposed method, Python/RDKit); Frag (sampling): assembly of molecular fragments using molecular graph operations (Python/RDKit); Smi: fast assembly of SMILES fragments (proposed method, OCaml). Dsmi: fast assembly of DeepSMILES fragments (proposed method, OCaml); Tag is the model training prerequisite of Smi and Dsmi sampling. All methods use a single CPU, except RNN which uses four CPUs and one GPU

Results

To assess the model training speed and molecular generation frequency of the proposed method, the 1M GDB-13 molecules sample [44] from Arús-Pous [8] was used and 2B molecules were generated (same protocol). The training set was fragmented (with a fragment molecular weight decreased from 150 to 50Da, because GDB-13 molecules are quite small) , then molecules were generated from those fragments (Fig. 4). Tests were run using a single core of an Intel Core i7 CPU @ 2.7GHz, with 12 cores and 16GB or RAM, under Ubuntu Linux 20.04 LTS.

To assess diversity of the generated molecules, as well as training set distribution matching, a sample of 100k molecules from ChEMBL-28 and a sample of 20k molecules from TCM@Taiwan were used. After molecular fragmentation of each set, the same number of molecules was generated (100k and 20k). The number of unique Bemis-Murcko scaffolds in each training and generated set is reported, along with the number at the intersection of those sets (Table 1).

To assess if the method is capable of training set distribution matching, those training and generated sets were projected into an eight dimensional space, where dimensions are quite unrelated (molecular weight, calculated LogP, #aromatic rings, topological polar surface area, #rotatable bonds, synthetic accessibility score [54], hydrogen bond acceptors and hydrogen bond donors). Then, the overlap between the training set and the corresponding generated set histogram was quantified using the Jaccard index (equation (1)). Let X and Y be two histograms with the same number of bins (n).

$$J(X,Y) = \frac{\sum_1^n \min(X_i, Y_i)}{\sum_1^n \max(X_i, Y_i)} \quad (1)$$

A Jaccard index of zero means no overlap between two histograms, while one means perfect similarity (Fig. 5).

Results on the GuacaMol molecular generation benchmark can be seen in Table 2 and Fig. 6.

Discussion

Model training frequency (Fig. 4). RNN [8] is the slowest method, with ~30 molecule/s (70 epochs; 8 min/epoch; 1M training set molecules). Plus, RNN used four CPUs and one GPU, so the per CPU processing frequency is much lower. Molecular fragmentation (2755 molecule/s) or cleaved bond tagging (proposed method; 2479 molecule/s); both written in Python using RDKit and running on a single CPU core have a more reasonable, and comparable, processing frequency.

Model sampling frequency (Fig. 4). Assembly of molecular fragments in Python using RDKit is the slowest method here (~1685 molecule/s). Editing a molecular graph (RWMol class in RDKit) is not so efficient. RNN is reasonably fast upon model sampling (~9167 molecule/s); although requiring four CPUs and one GPU. On the other hand, the proposed method of fast assembly

Table 1 Molecular diversity assessed via the number of unique Bemis-Murcko scaffolds at the intersection between datasets

	ChEMBL_train (100k)	ChEMBL_gene (100k)	TCM_train (20k)	TCM_gene (20k)
ChEMBL_train (100k)	25135	8466 (23957 new ~= 74%)	979	982
ChEMBL_gene (100k)	8466 (23957 new ~= 74%)	32423	802	891
TCM_train (20k)	979	802	6056	2713 (5726 new ~= 68%)
TCM_gene (20k)	982	891	2713 (5726 new ~= 68%)	8439

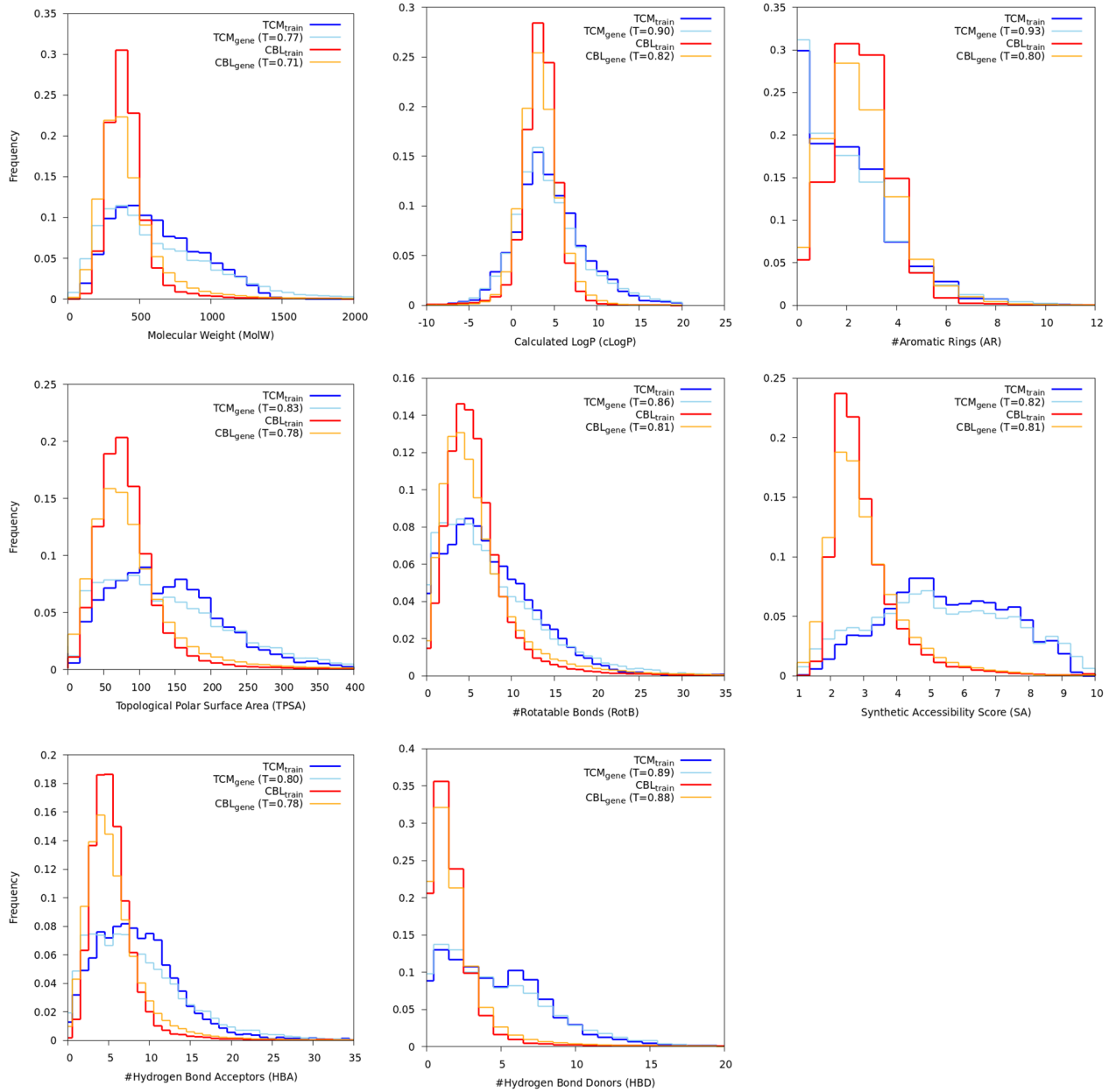

Fig. 5 Distribution reproduction experiments' histograms. Traditional Chinese Medicine database at Taiwan: training set in blue (TCM_{train}), generated set in cyan (TCM_{gene}). ChEMBL training set in red (CBL_{train}), generated set in orange (CBL_{gene}). The Jaccard index between the training and generated set histograms is shown in the legend (T = x)

of SMILES fragments reaches high sampling frequencies (~300599 SMILES-encoded molecule/s; ~348636 for DeepSMILES).

GuacaMol benchmark (Table 2) In this benchmark, while FASMIFRA is one of the simplest methods (probably just after the Random sampler baseline model), a balanced performance profile is observed. As expected, FASMIFRA generates only valid molecules (Validity = 1.0). The only metric in which FASMIFRA is not great is Novelty (0.7); meaning that sometimes generated molecules are part of the training set. But, FASMIFRA being a fragment-based method, this was expected. Especially, extended bond typing constrains which fragment can be connected to which, effectively limiting the number of combinations which can be obtained from a fragment library. On the other hand, a negative

Table 2 Comparison of several molecular generators in the GuacaMol [33] distribution learning benchmark

Benchmark	Random sampler	SMILES LSTM	Graph MCTS	AAE	ORGAN	VAE	FASMIFRA	Negative control
Validity	1.000	0.959	1.000	0.822	0.379	0.870	1.000	1.000
Uniqueness	0.997	1.000	1.000	1.000	0.841	0.999	0.994	0.959
Novelty	0.000	0.912	0.994	0.998	0.687	0.974	0.702	0.947
KL_divergence	0.998	0.991	0.522	0.886	0.267	0.982	0.959	0.855
FCD	0.929	0.913	0.015	0.529	0.000	0.863	0.814	0.397

Random sampler: baseline model; SMILES LSTM: Long-Short-Term Memory DNN for SMILES strings; Graph MCTS: Graph-based Monte Carlo Tree Search; AAE: Adversarial AutoEncoder; ORGAN: Objective-Reinforced Generative Adversarial Network; VAE: Variational AutoEncoder; FASMIFRA: Fast Assembly of SMILES Fragments (proposed method); Negative control: FASMIFRA without extended bond typing (any fragment can be connected to any other fragment)

control experiment was performed, where extended bond typing was turned off, which showed that while doing this improves the Novelty metric (from 0.702 to 0.947), this is at the detriment of training set distribution matching (KL_divergence is decreased from 0.959 to 0.855; FCD is decreased from 0.814 to 0.397). This negative control experiment shows that FASMIFRA is not a random molecular generator. Other methods which perform very well in the GuacaMol benchmark are the Random sampler baseline, but it cannot generate new molecules (Novelty = 0.0). The SMILES LSTM and the VAE are also very balanced and show good performance across all metrics. Compared to FASMIFRA, the Graph MCTS is lacking on the KL_divergence (0.522) and FCD metrics (0.015). The AAE is lacking on the FCD metric (0.529). The ORGAN is lacking on the Validity (0.379), KL_divergence (0.267) and FCD metrics (0.0). However, and to their defense, some of these methods might perform molecular optimization while FASMIFRA cannot (it would need to be coupled to a genetic algorithm or simulated annealer).

Molecular diversity (Table 1 and Novelty line in Table 2). In terms of Bemis-Murcko scaffolds, the proposed method generates a significant fraction of new scaffolds (74% in the generated set for ChEMBL; 68% in the generated set for TCM@Taiwan). With both datasets, generating molecules resulted in an increase of the number of unique Bemis-Murcko scaffolds in the generated set, compared to the training set (Table 1). As expected, the ChEMBL dataset (drug-like molecules) and the TCM@Taiwan dataset (natural products) are very different (Table 1 and Fig. 5). For example, both training sets share less than 1000 scaffolds.

Training set distribution matching (Fig. 5 and KL_divergence and FCD lines in Table 2). The method shows some propensity at training set distribution matching. For example, the minimum, median and maximum Jaccard indexes between histograms are (0.71, 0.805 and 0.88) for ChEMBL and (0.77, 0.845 and 0.93) for TCM@Taiwan.

On the positive side, this method is conceptually simple. Fragmenting molecules is reasonably fast (left of Fig. 4). Indexing fragments and generating molecules is extremely fast (right of Fig. 4). Only syntactically valid (Deep)SMILES encoded molecules are generated (Validity line in Table 2).

On the negative side, this method does require a training set. See the introduction for methods which don't. Also, the method doesn't create new rings. However, a medicinal chemistry technique is readily applicable in order to reasonably alter the generated rings [55]. If the training set contains molecules which cannot be fragmented (e.g. only made of fused rings), such molecules only contribute one seed fragment, without any attachable branch fragment (i.e. they might be copied as is from the training set to the generated set upon molecular generation). The method can generate duplicate molecules. Canonicalizing the produced SMILES would allow to detect and eventually filter those out.

Conclusion

In this article, a simple method to generate molecules from molecular fragments was described. The method can work with any molecular fragmentation scheme, as long as rings are not opened/broken. Several experiments were presented, evaluating model training speed, molecular generation frequency, molecular diversity and training set distribution matching. The proposed method can be used as-is in genetic algorithm or simulated annealing fragment-based molecular generators. Our prototype software implementation is released under the GPL license. The technique may also be useful for dataset augmentation and demonstrates that DeepSMILES proposed useful simplifications to the SMILES syntax. We are not working on it and it might be difficult, but an interesting extension might be to support molecular fragmentation schemes which happen to open/break rings.

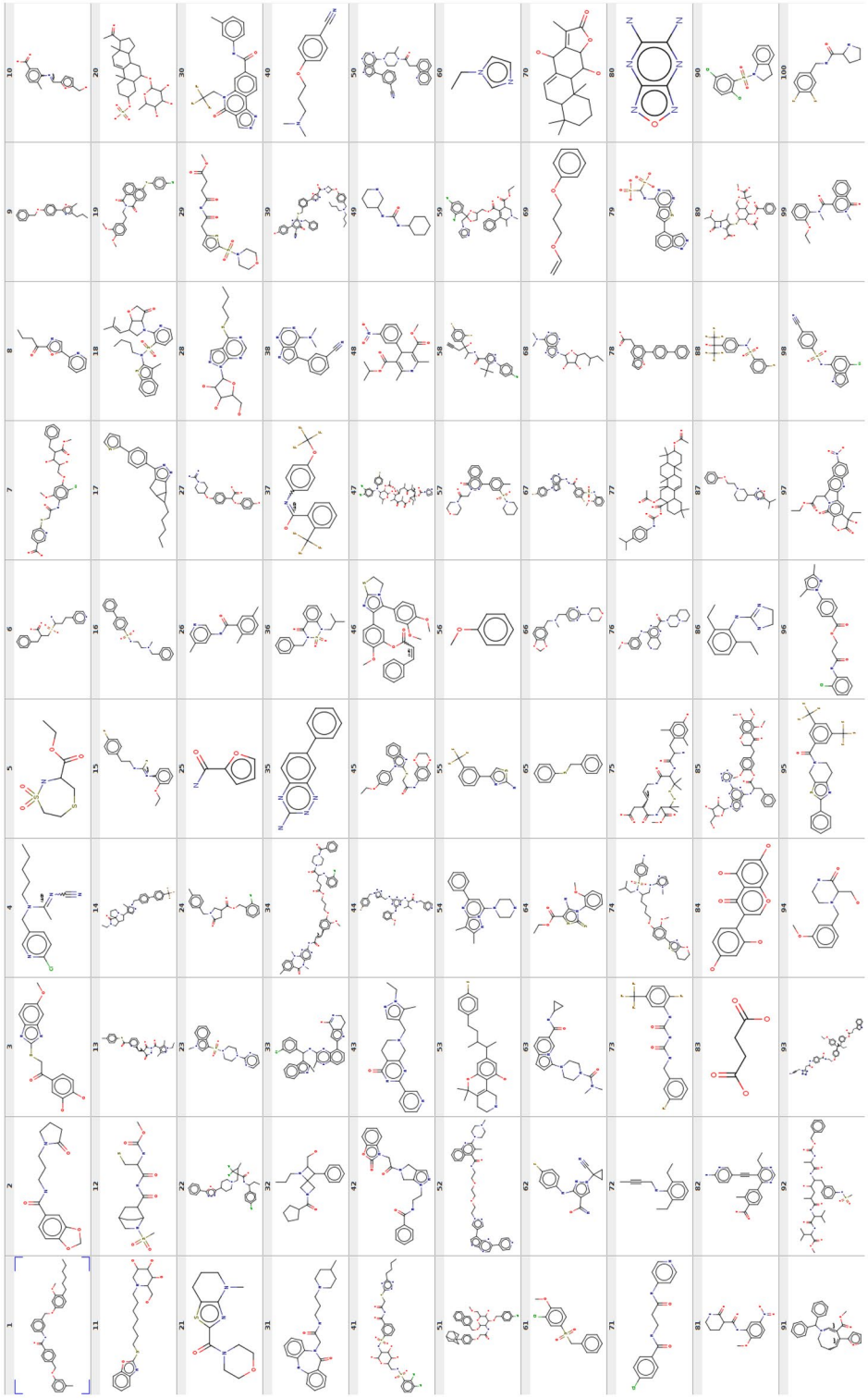

Fig. 6 100 FASMIFRA-generated molecules during the GuacaMol benchmark (ChEMBL 24 training set)

Abbreviations
AAE: Adverserial AutoEncoder; DNN: Deep Neural Network; FASMIFRA: Fast Assembly of SMILES Fragments; GDB: Generated DataBase; LSTM: Long Short-Term Memory; MCTS: Monte Carlo Tree Search; ORGAN: Objective-Reinforced Generative Adversarial Network; RNN: Recurrent Neural Network; SELFIES: Self-referencing embedded strings [28]; SMILES: Simplified Molecular Input Line Entry System [29]; VAE: Variational AutoEncoder.

Acknowledgements
FB acknowledges the use of ChemAxon JChem 20.13 for 2D depiction of molecules (https://chemaxon.com; accessed 2021-07-08).

Authors' contributions
FB designed and implemented the method, ran experiments, prepared figures and tables and wrote the initial manuscript. All authors have analyzed the results and participated in writing the manuscript. All authors read and approved the final manuscript.

Funding
This work was supported by the Japan Agency for Medical Research and Development: AMED [*JP*20*nk*0101111].

Availability of data and materials
The FASMIFRA software, TCM@Taiwan and ChEMBL training samples are on github: https://github.com/UnixJunkie/FASMIFRA (accessed 2021-07-28). The GDB-13 1M training sample can be downloaded from the GDB website [44, 45].

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing financial interests.

Received: 25 August 2021 Accepted: 2 November 2021
Published online: 14 November 2021

References

1. Grisoni F, Moret M, Lingwood R, Schneider G (2020) Bidirectional molecule generation with recurrent neural networks. J Chem Inf Model. 60(3):1175–1183. https://doi.org/10.1021/acs.jcim.9b00943
2. Gómez-Bombarelli R, Wei JN, Duvenaud D, Hernández-Lobato JM, Sánchez-Lengeling B, Sheberla D, Aguilera-Iparraguirre J, Hirzel TD, Adams RP, Aspuru-Guzik A (2018) Automatic chemical design using a data-driven continuous representation of molecules. ACS Cent Sci. 4(2):268–276. https://doi.org/10.1021/acscentsci.7b00572
3. Segler MHS, Kogej T, Tyrchan C, Waller MP (2018) Generating focused molecule libraries for drug discovery with recurrent neural networks. ACS Cent Sci. 4(1):120–131. https://doi.org/10.1021/acscentsci.7b00512
4. Neil D, Segler M, Guasch L, Ahmed M, Plumbley D, Sellwood M, Brown N (2018) Exploring deep recurrent models with reinforcement learning for molecule design. ICLR
5. Popova M, Isayev O, Tropsha A (2018) Deep reinforcement learning for de novo drug design. Adv Sci. https://doi.org/10.1126/sciadv.aap7885
6. ...Zhavoronkov A, Ivanenkov YA, Aliper A, Veselov MS, Aladinskiy VA, Aladinskaya AV, Terentiev VA, Polykovskiy DA, Kuznetsov MD, Asadulaev A, Volkov Y, Zholus A, Shayakhmetov RR, Zhebrak A, Minaeva LI, Zagribelnyy BA, Lee LH, Soll R, Madge D, Xing L, Guo T, Aspuru-Guzik A (2019) Deep learning enables rapid identification of potent DDR1 kinase inhibitors. Nat Biotechnol. 37(9):1038–1040. https://doi.org/10.1038/s41587-019-0224-x
7. Sattarov B, Baskin II, Horvath D, Marcou G, Bjerrum EJ, Varnek A (2019) De novo molecular design by combining deep autoencoder recurrent neural networks with generative topographic mapping. J Chem Inf Model. 59(3):1182–1196. https://doi.org/10.1021/acs.jcim.8b00751
8. Arús-Pous J, Blaschke T, Ulander S, Reymond J-L, Chen H, Engkvist O (2019) Exploring the GDB-13 chemical space using deep generative models. J Cheminf. 11(1):20. https://doi.org/10.1186/s13321-019-0341-z
9. Arús-Pous J, Johansson SV, Prykhodko O, Bjerrum EJ, Tyrchan C, Reymond J-L, Chen H, Engkvist O (2019) Randomized SMILES strings improve the quality of molecular generative models. J Cheminf. 11(1):71. https://doi.org/10.1186/s13321-019-0393-0
10. Blaschke T, Arús-Pous J, Chen H, Margreitter C, Tyrchan C, Engkvist O, Papadopoulos K, Patronov A (2020) Reinvent 2.0: an ai tool for de novo drug design. J Chem Inf Model. 60(12):5918–5922. https://doi.org/10.1021/acs.jcim.0c00915
11. Prykhodko O, Johansson SV, Kotsias P-C, Arús-Pous J, Bjerrum EJ, Engkvist O, Chen H (2019) A de novo molecular generation method using latent vector based generative adversarial network. J Cheminf. 11(1):74. https://doi.org/10.1186/s13321-019-0397-9
12. Arús-Pous J, Patronov A, Bjerrum EJ, Tyrchan C, Reymond J-L, Chen H, Engkvist O (2020) SMILES-based deep generative scaffold decorator for de-novo drug design. J Cheminf. 12(1):38. https://doi.org/10.1186/s13321-020-00441-8
13. Yang X, Zhang J, Yoshizoe K, Terayama K, Tsuda K (2017) ChemTS: an efficient Python library for de novo molecular generation. Sci Technol Adv Mater. 18(1):972–976. https://doi.org/10.1080/14686996.2017.1401424
14. Sumita M, Yang X, Ishihara S, Tamura R, Tsuda K (2018) Hunting for organic molecules with artificial intelligence: molecules optimized for desired excitation energies. ACS Central Sci 4(9):1126–1133. https://doi.org/10.1021/acscentsci.8b00213
15. Merk D, Friedrich L, Grisoni F, Schneider G (2018) De novo design of bioactive small molecules by artificial intelligence. Mol Inf. 37(1–2):1700153. https://doi.org/10.1002/minf.201700153
16. Gupta A, Müller AT, Huisman BJH, Fuchs JA, Schneider P, Schneider G (2018) Generative recurrent networks for de novo drug design. Mol Inf. 37(1–2):1700111. https://doi.org/10.1002/minf.201700111
17. Blaschke T, Olivecrona M, Engkvist O, Bajorath J, Chen H (2018) Application of generative autoencoder in de novo molecular design. Mol Inf. 37(1–2):1700123. https://doi.org/10.1002/minf.201700123
18. Liu X, Ye K, van Vlijmen H.W.T, IJzerman A.P, van Westen G.J.P (2019) An exploration strategy improves the diversity of de novo ligands using deep reinforcement learning: a case for the adenosine A2A receptor. J Cheminf 11(1):35. https://doi.org/10.1186/s13321-019-0355-6
19. Polishchuk P (2020) CReM: chemically reasonable mutations framework for structure generation. J Cheminf. 12(1):28. https://doi.org/10.1186/s13321-020-00431-w
20. Kwon Y, Lee J (2021) MolFinder: an evolutionary algorithm for the global optimization of molecular properties and the extensive exploration of chemical space using SMILES. J Cheminf. 13(1):24. https://doi.org/10.1186/s13321-021-00501-7
21. Nigam A, Pollice R, Krenn M, Gomes GdP, Aspuru-Guzik A (2021) Beyond generative models: superfast traversal, optimization, novelty, exploration and discovery (STONED) algorithm for molecules using SELFIES. Chem Sci. 12:7079–7090. https://doi.org/10.1039/D1SC00231G
22. Jensen JH (2019) A graph-based genetic algorithm and generative model/Monte Carlo Tree search for the exploration of chemical space. Chem Sci. 10(12):3567–3572. https://doi.org/10.1039/c8sc05372c
23. Brown N, McKay B, Gilardoni F, Gasteiger J (2004) A graph-based genetic algorithm and its application to the multiobjective evolution of median molecules. J Chem Inf Comp Sci. 44(3):1079–1087. https://doi.org/10.1021/ci034290p
24. Leguy J, Cauchy T, Glavatskikh M, Duval B, Da Mota B (2020) EvoMol: a flexible and interpretable evolutionary algorithm for unbiased de novo molecular generation. J Cheminf. 12(1):55. https://doi.org/10.1186/s13321-020-00458-z
25. Polishchuk P (2020) Control of synthetic feasibility of compounds generated with CReM. J Chem Inf Model. 60(12):6074–6080. https://doi.org/10.1021/acs.jcim.0c00792
26. Joung I, Kim JY, Gross SP, Joo K, Lee J (2018) Conformational space annealing explained: a general optimization algorithm, with diverse applications. Comput Phys Commun. 223:28–33. https://doi.org/10.1016/j.cpc.2017.09.028

27. Yoshikawa N, Terayama K, Sumita M, Homma T, Oono K, Tsuda K (2018) Population-based de novo molecule generation. Using grammatical evolution. Chem Lett. 47(11):1431–1434. https://doi.org/10.1246/cl.180665
28. Krenn M, Häse F, Nigam A, Friederich P, Aspuru-Guzik A (2020) Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. Mach Learn Sci Technol. 1(4):045024. https://doi.org/10.1088/2632-2153/aba947
29. Weininger D (1988) SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci. 28(1):31–36. https://doi.org/10.1021/ci00057a005
30. James CA, et al OpenSMILES specification. http://opensmiles.org/opensmiles.html. Accessed 7 July 2021
31. O'Boyle N, Dalke A (2018) DeepSMILES: an adaptation of SMILES for use in machine-learning of chemical structures. ChemRxiv. https://doi.org/10.26434/chemrxiv.7097960.v1
32. Bemis GW, Murcko MA (1996) The properties of known drugs. 1. Molecular frameworks. J Med Chem. 39(15):2887–2893. https://doi.org/10.1021/jm9602928
33. Brown N, Fiscato M, Segler MHS, Vaucher AC (2019) GuacaMol: benchmarking models for de novo molecular design. J Chem Inf Model. 59(3):1096–1108. https://doi.org/10.1021/acs.jcim.8b00839
34. Nathan B GuacaMol leaderbord. https://www.benevolent.com/guacamol. Accessed 23 Aug 2021
35. Nathan B GuacaMol github. https://github.com/BenevolentAI/guacamol. Accessed 23 Aug 202
36. Nathan B GuacaMol baselines github. https://github.com/BenevolentAI/guacamol_baselines. Accessed 23 Aug 202
37. Gaulton A, Hersey A, Nowotka M, Bento AP, Chambers J, Mendez D, Mutowo P, Atkinson F, Bellis LJ, Cibrián-Uhalte E, Davies M, Dedman N, Karlsson A, Magariños MP, Overington JP, Papadatos G, Smit I, Leach AR (2016) The ChEMBL database in 2017. Nucleic Acids Res. 45(D1):945–954. https://doi.org/10.1093/nar/gkw1074
38. Landrum G RDKit: Open-Source Cheminformatics. http://www.rdkit.org. Accessed 8 July 2021
39. Preuer K, Renz P, Unterthiner T, Hochreiter S, Klambauer G (2018) Fréchet ChemNet distance: a metric for generative models for molecules in drug discovery. J Chem Inf Model. 58(9):1736–1741. https://doi.org/10.1021/acs.jcim.8b00234
40. Kullback S, Leibler RA (1951) On information and sufficiency. Ann Math Stat. 22(1):79–86
41. Polykovskiy D, et al. MOSES. https://github.com/molecularsets/moses. Accessed 23 Aug 2021
42. Polykovskiy D, Zhebrak A, Sanchez-Lengeling B, Golovanov S, Tatanov O, Belyaev S, Kurbanov R, Artamonov A, Aladinskiy V, Veselov M, Kadurin A, Johansson S, Chen H, Nikolenko S, Aspuru-Guzik A, Zhavoronkov A (2020) Molecular sets (MOSES): a benchmarking platform for molecular generation models. Front Pharmacol. 11:1931. https://doi.org/10.3389/fphar.2020.565644
43. Guimaraes GL, Sanchez-Lengeling B, Outeiral C, Farias PLC, Aspuru-Guzik A (2018) Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models, 1–7. arXiv:1705.10843
44. Arús-Pous J GDB13 1M sample. http://gdbtools.unibe.ch:8080/cdn/gdb13.1M.freq.ll.smi.gz. Accessed 8 July 2021
45. Blum LC, Reymond J-L (2009) 970 million druglike small molecules for virtual screening in the chemical universe database gdb-13. JACS 131(25):8732–8733. https://doi.org/10.1021/ja902302h
46. Chen CY-C (2011) TCM Database@Taiwan: the world's largest traditional Chinese medicine database for drug screening in silico. PLoS ONE. 6(1):1–5. https://doi.org/10.1371/journal.pone.0015939
47. Degen J, Wegscheid-Gerlach C, Zaliani A, Rarey M (2008) On the art of compiling and using "Drug-Like" chemical fragment spaces. ChemMedChem 3(10):1503–1507. https://doi.org/10.1002/cmdc.200800178
48. Lewell XQ, Judd DB, Watson SP, Hann MM (1998) RECAP - Retrosynthetic combinatorial analysis procedure: a powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry. J Chem Inf Comput Sci. 38(3):511–522. https://doi.org/10.1021/ci970429i
49. Berenger F, Yamanishi Y (2020) Ranking molecules with vanishing kernels and a single parameter: active applicability domain included. J Chem Inf Model. 60(9):4376–4387. https://doi.org/10.1021/acs.jcim.9b01075
50. Carhart RE, Smith DH, Venkataraghavan R (1985) Atom Pairs as molecular features in structure-activity studies: definition and applications. J Chem Inf Comput Sci. 25(2):64–73. https://doi.org/10.1021/ci00046a002
51. Liu T, Naderi M, Alvin C, Mukhopadhyay S, Brylinski M (2017) Break down in order to Build up: decomposing small molecules for fragment-based drug design with eMolFrag. J Chem Inf Model. 57(4):627–631. https://doi.org/10.1021/acs.jcim.6b00596
52. Berenger F, Zhang KYJ, Yamanishi Y (2019) Chemoinformatics and structural bioinformatics in ocaml. J Cheminf. 11(1):10. https://doi.org/10.1186/s13321-019-0332-0
53. Leroy X, Doligez D, Frisch A, Garrigue J, Rémy D, Vouillon J (2021) The OCaml System Release 4.12 - Documentation and User's Manual. INRIA, Paris, France
54. Ertl P, Schuffenhauer A (2009) Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. J Cheminf. 1(1):8. https://doi.org/10.1186/1758-2946-1-8
55. Pennington LD, Aquila BM, Choi Y, Valiulin RA, Muegge I (2020) Positional analogue scanning: an effective strategy for multiparameter optimization in drug design. J Med Chem. 63(17):8956–8976. https://doi.org/10.1021/acs.jmedchem.9b02092

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions