Research Article
Malays. j. med. biol. res.

Molecular Generators and Optimizers Failure Modes

Mani Manavalan[*]

Sr. Manager, Capgemini America, 79 5th Avenue, Suite 300, New York, NY 10003, **USA**

***Email for Correspondence:** mani.manavalan@capgemini.com

ABSTRACT

In recent years, there has been an uptick in interest in generative models for molecules in drug development. In the field of de novo molecular design, these models are used to make molecules with desired properties from scratch. This is occasionally used instead of virtual screening, which is limited by the size of the libraries that can be searched in practice. Rather of screening existing libraries, generative models can be used to build custom libraries from scratch. Using generative models, which may optimize molecules straight towards a desired profile, this time-consuming approach can be sped up. The purpose of this work is to show how current shortcomings in evaluating generative models for molecules can be avoided. We cover both distribution-learning and goal-directed generation with a focus on the latter. Three well-known targets were downloaded from ChEMBL: Janus kinase 2 (JAK2), epidermal growth factor receptor (EGFR), and dopamine receptor D2 (DRD2) (Bento et al. 2014). We preprocessed the data to get binary classification jobs. Before calculating a scoring function, the data is split into two halves, which we shall refer to as split 1/2. The ratio of active to inactive users. Our goal is to train three bioactivity models with equal prediction performance, one to be used as a scoring function for chemical optimization and the other two to be used as performance evaluation models. Our findings suggest that distribution-learning can attain near-perfect scores on many existing criteria even with the most basic and completely useless models. According to benchmark studies, likelihood-based models account for many of the best technologies, and we propose that test set likelihoods be included in future comparisons.

Keywords: Molecular Generators, Molecular design, Optimizers failure modes, Generative models, Distribution learning, Goal-directed generation

Manuscript Received: 15 July 2021 - Revised: 28August 2021 - Accepted: 03 Sept 2021

INTRODUCTION

Segler et al. (2018); Gomez-Bombarelli et al. (2018); Sanchez-Lengeling and Aspuru-Guzik (2018) and Olivecrona et al. (2017) have all noted a rise in interest in generative models for molecules in drug discovery in recent years. Those models are used to produce molecules with desired attributes from scratch in the field of de novo molecular design (Schneider 2013). This is sometimes considered as a replacement for virtual screening, which is constrained by the size of the libraries that can be searched in practice. Rather than screening existing libraries, generative models can be used to construct targeted libraries from the ground up (Segler et al. 2018). They can also help with lead optimization, which entails finding a small number of compounds with optimal profiles. Potency, metabolic stability, physicochemical characteristics, and permeability are just a few of the dimensions that can be included in a profile (Bynagari, 2016). This time-consuming procedure can be sped up by using generative models, which can optimize molecules straight towards a desired profile (Donepudi, 2018).

Machine learning (ML), specifically Deep Learning, is used to create new generative models for molecules (Schmidhuber 2015; LeCun, Bengio, and Hinton 2015). More traditional approaches (Venkatasubramanian, Chan, and Caruthers 1994; Douguet, Thoreau, and Grassy 2000; Jensen 2019) are complemented by these. The capacity of machine learning technologies to learn what viable compounds look like from data is their key advantage over traditional approaches. To achieve this, machine learning methods employ a training set of molecules from which they attempt to deduce the data's underlying distribution. The learned distribution is then utilized to create new molecules, with different approaches differing in the specifics of how the creation process is carried out.

The SMILES (Weininger 1988) representation of chemicals was utilized in tandem with recurrent neural networks (RNNs) (Hochreiter and Schmidhuber 1997; Cho et al. 2014) in the first wave of Deep Learning methods for molecule production (Segler et al. 2018; Gomez-Bombarelli et al. 2018).

More recent versions of these models, such as DeepSmiles (O'Boyle and Dalke 2018) or SELFIES (Krenn et al. 2020), have progressed toward more robust line notations of chemical structure. Graph neural networks are used in another set of models to directly build molecular graphs (Scarselli et al. 2009). Models differ in terms of training technique and model architecture, in addition to the representation used. See (Sanchez-Lengeling and Aspuru-Guzik 2018; Elton et al. 2019) for a more extensive discussion of current techniques. Models differ in terms of training technique and model architecture, in addition to the representation used. See (Sanchez-Lengeling and Aspuru-Guzik 2018; Elton et al. 2019) for a more extensive discussion of current techniques.

Distribution-learning and goal-directed generation are the two most common applications of generative models for molecules (Brown et al. 2019). The task of producing molecules that look like a given set of molecules in distribution is dealt with by distribution-learning. The goal of goal-directed generation is to create molecules with a certain set of qualities, such as physical/chemical properties, bioactivities, or a combination of these (Fadziso & Manavalan, 2017; Manavalan, 2016).

Models based on distribution-learning can be used to produce chemical libraries or as a jumping off point for goal-directed generation. Recent efforts to set benchmarks for their evaluation (Preuer et al. 2018; Brown et al. 2019; Polykovskiy et al. 2018) have shown that evaluating these models can be difficult. These benchmarks frequently include algorithms that attempt to capture desired aspects of the created molecules, such as comparing the distributions of different chemical attributes. However, simple models can fool many of these heuristics.

Goal-directed generative models for molecules are trained to generate molecules with a specific property profile, such as physical or chemical properties, bioactivities, or a combination of the three. Kusner, Paige, and Hernandez-Lobato (2017) assessed a number of such models for their capacity to generate compounds with a high penalized logP score (You et al. 2019; Zhou et al. 2018; Zhang et al. 2019). It is important to highlight, however, that producing lengthy saturated hydrocarbon chains is a simple way to attain state-of-the-art outcomes. Brown et al. (2019) developed the GuacaMol package, which helped to improve evaluation. However, we agree with the authors that the benchmarks are simple to solve and that the quality of the compounds generated is not well addressed (Manavalan, 2018).

A scoring function is usually used in conjunction with a goal-directed molecule generator to determine how closely a molecule matches the desired profile. Unfortunately, developing a decent scoring function for many jobs is difficult, because this function must assess a molecule's biological effects as well as synthetic feasibility and drug-likeness, all of which are difficult to define (Bickerton et al. 2012; Ertl and Schuffenhauer 2009). The majority of scoring functions used in practice do not take into account practitioners' intuitive limits. As a result, a molecular generator may optimize the scoring function in ways that were not intended. When machine learning models are used as scoring functions, the problem becomes even worse.

Despite a flood of papers describing innovative approaches to molecule creation, wet-lab validations of generative models are still rare. In vitro activity of compounds created using transfer learning (Segler et al. 2018) was demonstrated by Merk, Friedrich, et al. (2018) and Merk, Grisoni, et al. (2018). Using generative Deep Learning models, Zhavoronkov et al. (2019) found a DDR1 kinase inhibitor. What these studies don't show is how "innovative" the proposed approaches are, or how well they would perform in compared to chemists designing compound libraries using more traditional methods. We anticipate that as the field evolves, such comparisons will become increasingly common.

Objectives of the Study

The goal of this paper is to illustrate present flaws in evaluating generative models for molecules. With a focus on the latter, we address both distribution-learning and goal-directed generation.

LITERATURE REVIEW

Mechanisms of failure in distribution-learning

First, we examine the shortcomings of existing metrics for distribution-learning and show that many of them can be easily fooled by a model that only modifies the molecules in the training set little.

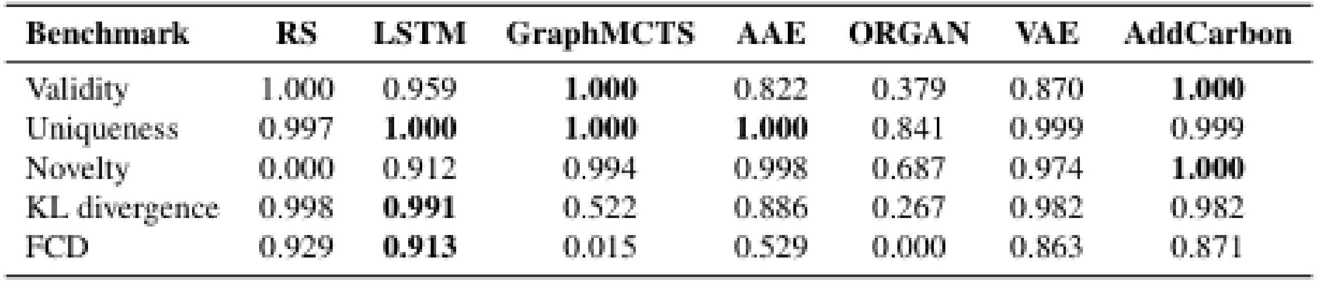
Segler et al. (2018) and Li et al. (2018), variational auto-encoders (Gomez-Bombarelli et al. 2018; Kusner, Paige, and Hernandez-Lobato 2017; Jin, Barzilay, and Jaakkola 2018), generative adversarial networks (Guimaraes et al. 2017; Sanchez-Lengeling, Outeiral, et al. 2018), and generative adversarial previously, only a few criteria were used to evaluate performance in distribution-learning studies, such as visual inspection, novelty, and validity. Checking the originality of generated compounds is the most common method for identifying duplication of molecules in the training set. Because it only looks for exact compound matches in the training set, this statistic is relatively insensitive. The validity metric determines if a synthesized molecule has the correct syntactic structure. Uniqueness is also determined if molecules appear multiple times in a group of generated molecules. The Frechet Chemnet Distance (FCD) was established by Preuer et al. (2018) and has been demonstrated to incorporate a range of heuristics in a single score. The FCD is included in more comprehensive benchmarking suites (Brown et al. 2019; Polykovskiy et al. 2018). While the molecules produced by distribution-learning models may appear clear, objectively determining whether the model grasped existing patterns in the data distribution or merely duplicated the training inputs is difficult.

In general, evaluating generative models is difficult and necessitates the creation of new metrics (Preuer et al. 2018; Brown et al. 2019; Polykovskiy et al. 2018). These measurements, on the other hand, are unable to determine when algorithms repeat the training data with minimum alterations, which we refer to as the copy problem.

To demonstrate the copy problem, we show how a simple model called AddCarbon can fool most distribution learning measures (Brown et al. 2019). To sample a "new" molecule, our model selects a random molecule from the training set. Then, in its SMILES representation, a carbon atom is introduced at a random location. If a syntactically valid SMILES and a molecule not previously in the training set are returned, a new random sample is generated. Other insert places are explored if the carbon atom insertion results in an invalid SMILES string. If none of the positions work, a new molecule is picked from the training set, and the processes are repeated until success is achieved.

The GuacaMol distribution learning benchmark is used to assess this model (Brown et al. 2019). The AddCarbon model outperforms many complicated generative models and achieves near-perfect benchmarking scores (see Table 1). It possesses a 100 percent originality and validity by construction, as well as almost perfect uniqueness and a very high Kullback–Leibler (KL) divergence measure.

Table 1: Comparison of the AddCarbon model to the baselines in (Brown et al. 2019)



Benchmark	RS	LSTM	GraphMCTS	AAE	ORGAN	VAE	AddCarbon
Validity	1.000	0.959	**1.000**	0.822	0.379	0.870	**1.000**
Uniqueness	0.997	**1.000**	**1.000**	**1.000**	0.841	0.999	0.999
Novelty	0.000	0.912	0.994	0.998	0.687	0.974	**1.000**
KL divergence	0.998	**0.991**	0.522	0.886	0.267	0.982	0.982
FCD	0.929	**0.913**	0.015	0.529	0.000	0.863	0.871

The FCD was the only metric for which a competitive score was difficult to come by. This was unexpected because the FCD is dependent on the SMILES representation, which was forced to be similar to those in the training set in our naive model expressly to exploit the FCD. Except for the LSTM model, we were able to beat all of the baselines with this simple model. The fact that the simple AddCarbon model is useless in practice while yet scoring well raises questions about whether the measures currently in use are adequate for estimating performance.

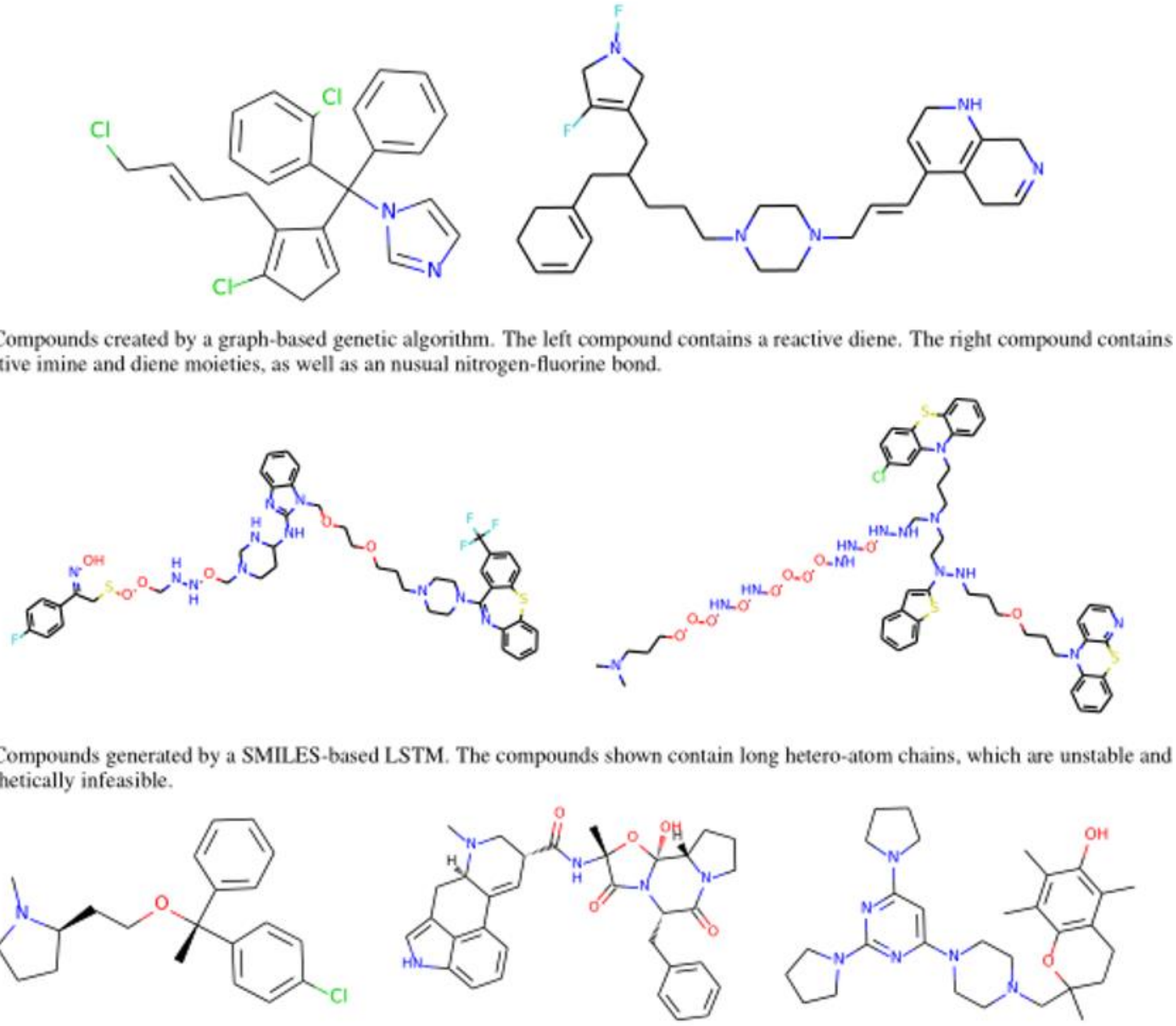
Our tests demonstrate that a more precise measurement of novelty would be extremely beneficial. While the FCD recognizes that these naively created molecules do not match the training distribution, a naive model similar to the AddCarbon model might also fool it. The measurements currently in use do not allow us to determine whether the molecule generators behave similarly to our AddCarbon model. Many of the best techniques in the GuacaMol (Brown et al. 2019) and Moses (Polykovskiy et al. 2018) benchmarks are likelihood-based models (Segler et al. 2018; Jin, Barzilay, and Jaakkola 2018; Gomez-Bombarelli et al. 2018). Similar to natural language processing, these performances could be evaluated using the likelihood on a hold-out test set. We believe that if this parameter is relevant, it should be reported in future studies (Manavalan & Donepudi, 2016).

Mechanisms of failure in goal-directed generation

We examine probable failure modes of goal-directed molecule producers under this subheading. The construction of a scoring function that encompasses all of the desired features a molecule should have in a robust fashion, we suggest, is a central difficulty in this scenario. We show that using predictions from a machine learning model as part of the score exacerbates the situation.

Goal-directed generation focuses on discovering molecules that optimize a desired scoring function that must capture the task's requirements (Bynagari, 2017). This process is difficult in and of itself because it is difficult to condense complicated chemical qualities into a single number, and it becomes even more difficult when attempting to optimize for many properties at the same time.

Even if the created molecules achieve high scores, they may do so in ways that the practitioner does not anticipate. While the challenge can be regarded solved from an optimization standpoint if high-scoring molecules are produced, the outcomes may not be satisfactory or helpful. For example, the produced molecules in Figure 1 have high scores yet contain substructures that are unstable or synthetically infeasible.



(a) Compounds created by a graph-based genetic algorithm. The left compound contains a reactive diene. The right compound contains reactive imine and diene moieties, as well as an nusual nitrogen-fluorine bond.

(b) Compounds generated by a SMILES-based LSTM. The compounds shown contain long hetero-atom chains, which are unstable and synthetically infeasible.

(c) Reference compounds from the training set

Figure 1: High scoring compounds generated for the DRD2 task described below, with actives from the training set for comparison

In practice, we've discovered that generative models are really good at creating unexpected solutions that are numerically superior but not very useful (Bynagari, 2018). Before arriving at compounds that could be relevant for drug development efforts, this generally necessitates numerous repetitions between creating molecules and adjusting the scoring function to account for previously unforeseen behavior from the generator. We've noticed that as more machine learning models are added to the scoring function, this tendency becomes even more pronounced.

Other applications (Lehman et al. 2019) have observed this tendency of score optimization in unexpected ways. The goal in one example was to create a locomotion-capable body, but the optimization technique instead revealed the simpler solution of a tall body falling over, which also satisfied the scoring function.

Exact scoring functions are not accessible for many jobs in drug discovery. While some qualities, such as molecular mass, may be estimated with precision given a compound, more complicated properties, such as bioactivities, cannot. Instead, machine learning models are frequently fitted to experimental data to approximate these (Olivecrona et al. 2017; Popova et al., 2018; Segler et al. 2018; Neogy & Bynagari, 2018). These models can then be utilized as a scoring function or as a component of a scoring function.

Machine learning models include biases based on the data they were trained on as well. Models frequently show near-perfect prediction performance on training data, but perform poorly on hold-out data (Bynagari & Fadziso, 2018). This is accomplished by constructing predictions based on fictitious patterns that can be utilized to link samples to labels but aren't actual explanatory aspects of the output. As a result, these erroneous patterns may be retrieved when a learnt model's outputs are improved (Donepudi, 2017). Furthermore, it is usually observed that actives/inactives from the training set get higher/lower prediction scores than those who took the test set. This could lead to a bias in the formation of chemical scores, similar to how they are skewed in the training set actives that are in favor. Prejudices based on data are referred to as data-specific prejudices.

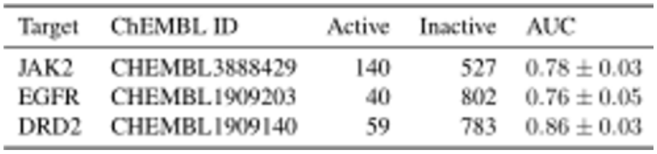
Methods

To demonstrate these failure patterns, we will now present our experimental setup. Our goal is to create compounds that are capable of attacking a biological target. We downloaded data from ChEMBL for three well-known targets: Janus kinase 2 (JAK2), epidermal growth factor receptor (EGFR), and dopamine receptor D2 (DRD2) (Bento et al. 2014). To get binary classification tasks, we preprocessed the data. Table 2 contains information regarding the data. The data is split into two halves, which we will refer to as split 1/2, before calculating a scoring function. The percentage of actives versus inactives. Our goal is to train three bioactivity models with similar prediction performance, one of which will be utilized as a scoring function for molecular optimization and the other two as performance evaluation models. Three classifiers are trained to achieve this goal. The first classifier, which was trained on split 1, will be utilized as a scoring function for optimizing molecules, and its output will be called optimization score (OS).

The second classifier uses a different random seed than the OS model to train on split 1. This classifier assesses model specific biases and controls whether the scores of optimized molecules generalize across two classifiers trained on the same data. Model control scores are the outputs of the model (MCS). The third classifier, which was learned on split 2, is used to see if the optimized molecules perform well when compared to a model that was trained on other samples and quantifies data-specific biases. The data control score is the name given to the result of this classifier (DCS).

As a classification approach, we employ a random forest classifier (Breiman 2001) implemented in scikit-learn (Pedregosa et al. 2011). As a scoring function, the ratio of trees predicting that a molecule is active is employed. We employ binary folded ECFP fingerprints of size 1024 and radius 2 (Rogers and Hahn 2010) as features, which were calculated using rdkit (Landrum 2006). For each of the three targets, JAK2, EGFR, and DRD2, we found three classifiers with equivalent predictive performance (see Table 2) that are acceptable for goal-directed generation. The performance of each model was assessed on the split that was not utilized for training. Because the performance of all three classifiers is expected to be independent of the data split and random seed used, there is only one performance number for all three.

Table 2: Information on the data sets



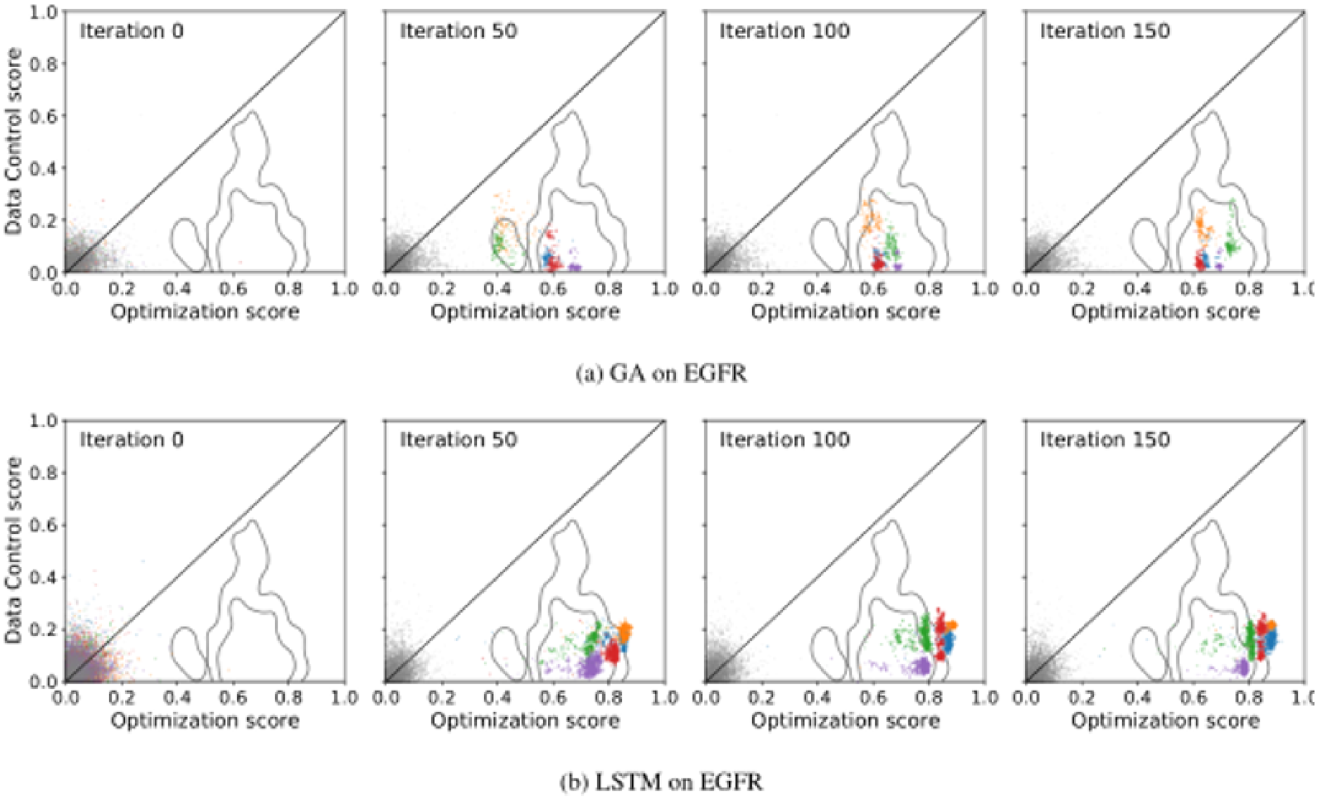
Target	ChEMBL ID	Active	Inactive	AUC
JAK2	CHEMBL3888429	140	527	0.78 ± 0.03
EGFR	CHEMBL1909203	40	802	0.76 ± 0.05
DRD2	CHEMBL1909140	59	783	0.86 ± 0.03

Then, using a goal-directed generation technique, we were able to generate molecules with high optimization scores. According to Guacamol, the two best-performing molecular generators are used here (Brown et al. 2019). A graph-based genetic algorithm (GA) is one of them (Jensen 2019). GA improves molecules by applying random mutations and crossovers to a population of molecules and keeping the best in each generation. Random molecules from the distribution-learning training set defined in (Brown et al. 2019), which is a random subset of the chemicals in ChEMBL (Bento et al. 2014), were used as the beginning population. SMILES-LSTM (LSTM) is the second optimization algorithm we apply (Segler et al. 2018). This approach produces molecules by inferring the probabilities of the next character in SMILES strings based on the probabilities of the previous ones (Donepudi, 2016). The molecules are fine-tuned using a hill climbing method, which involves sampling molecules iteratively, keeping the best ones, and fine-tuning the model on these high-scoring molecules. To initialize the generator, we used the pretrained model given by (Brown et al. 2019). The starting population for GA was taken from the same data set that this model was trained on. For each data set, we perform each optimization technique ten times.

The proposed experimental approach, which includes optimization and control scores, allows us to learn how a generative model optimizes the score and whether it is affected by the bioactivity model's biases. A training and test set are used in supervised learning methods, and the setup with an optimization and control score is similar (Donepudi, 2015). The purpose of supervised learning is to achieve good results on a test set that was not used during optimization, which corresponds to the control scores in our case (Manavalan & Bynagari, 2015).

Results and Discussion

First, we look at how the optimization score (OS) and data control score (DCS) change during EGFR task optimization (see Figure 2). We present the scores of molecules in the population at various iterations for GA, but we sample molecules in each iteration for LSTM. The optimization procedure boosts the OS more than the DCS when starting with random ChEMBL compounds. The DRD2 and JAK2 data sets exhibit the same behavior. There is a mismatch between OS and DCS for all of the jobs addressed here, indicating that the optimization technique has model and/or data specific biases.



(a) GA on EGFR

(b) LSTM on EGFR

Figure 2: Scatter plots of OS vs. DCS on molecules during the course of training

As evidenced by their migration towards the outlines in the plot, the optimized molecules appear to inhabit the same region as the split 1 actives that were used to train the OS. This indicates that the molecular optimizer looks for compounds that are related to the actives used to calculate the optimization scoring function. As determined by ECFP4 Tanimoto similarity, this is really the case, as optimized molecules have a more comparable neighbor in split 1 than in split 2. This demonstrates that data particular biases can account for at least some of the difference between OS and DCS.

Next, we look at how much of the discrepancy between OS and DCS is due to model and data biases. Figure 3 depicts the evolution of all three scores over the period of optimization. In the process of training, we notice that the OS and MCS diverge. This shows that, despite being trained on the identical data, optimization uses properties unique to the OS to obtain improvements that do not generalize to the MCS. It's also worth noting that data-specific biases account for the majority of the difference between OC and DCS.

Figure 3 further illustrates that, while the OS rises steadily, the control scores occasionally plateau or even decline. As a result, optimization should be stopped as soon as the control scores stop increasing.

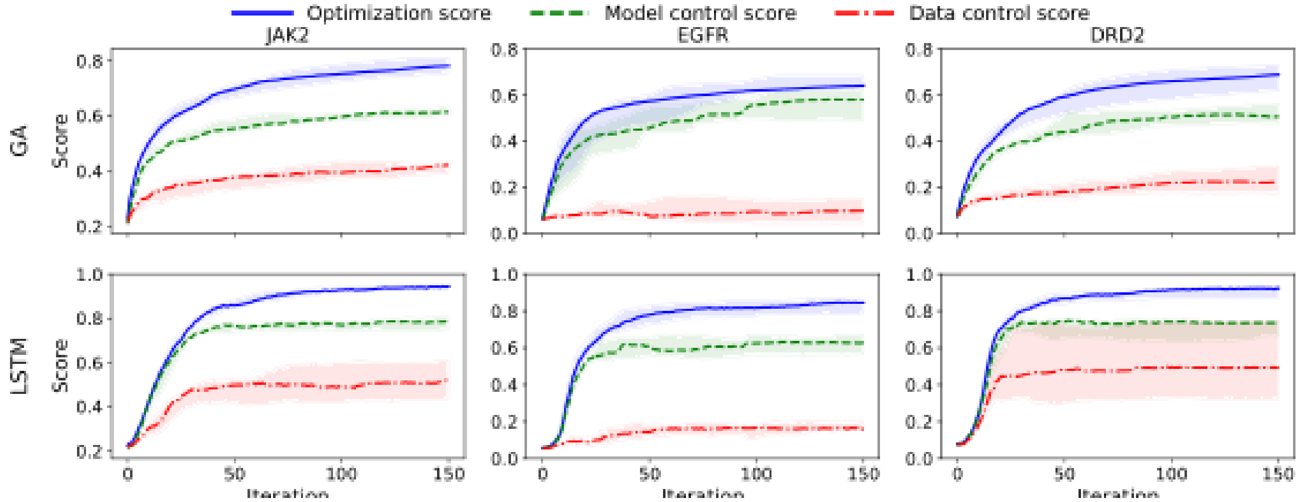


Figure 3: Scores during optimization. For each curve we first took the mean of the scores for each run

The design decisions we made constrained our trials and the conclusions we drew from them. The LSTM and GA molecular optimizers were used because they worked effectively in GuacaMol (Brown et al. 2019). However, it's possible that other algorithms don't have the same flaws as the ones described above. We went with ECFP fingerprints because they've been demonstrated to work well in the past (Mayr et al. 2018). Random Forest classifiers were chosen because they are reasonably easy to train and have good performance (Donepudi, 2014). When using different data sets, the results may differ. Future investigations will, however, focus on more thorough experimentation.

Conclusion and Recommendation

We looked into generative models for molecules in light of existing evaluation methodologies in this paper. Our conclusion on distribution-learning is that even the most basic and practically useless models can achieve near-perfect scores on many current criteria. Many of the best technologies, according to benchmark studies (Brown et al. 2019; Polykovskiy et al. 2018), are likelihood-based models, and we propose that test set likelihoods should be included in future comparisons. This would be a complete gauge that would be an improvement over previously suggested measurements that may be “tricked.” We talked about different failure modes in goal-directed learning, with a focus on issues that arise when machine learning models are used as scoring functions. While various technological tools and strategies can be used to improve a scoring function, the scoring function itself is the main challenge. We demonstrated that optimization takes advantage of model-based scoring functions' data and model-specific biases. Given that the goal of generative models for de novo design is to explore all of chemical space, biases toward training data could imply a failure in this regard.

References

Bento, A. P., Gaulton, A., Hersey, A., Bellis, L. J., Chambers, J., Davies, M., Kruger, F. A., Light, Y., Mak, L., McGlinchey, S., et al. (Jan. 1, 2014). The ChEMBL Bioactivity Database: An Update. In: Nucleic Acids Res 42.D1, pp. D1083–D1090.

Bickerton, G. R., Paolini, G. V., Besnard, J., Muresan, S., and Hopkins, A. L. (Jan. 24, 2012). Quantifying the Chemical Beauty of Drugs. In: Nat Chem 4.2, pp. 90– 98.

Breiman, L. (Oct. 1, 2001). Random Forests. In: Mach Learn 45.1, pp. 5–32.

Brown, N., Fiscato, M., Segler, M. H., and Vaucher, A. C. (Mar. 25, 2019). GuacaMol: Benchmarking Models for de Novo Molecular Design. In: J Chem Inf Model 59.3, pp. 1096–1108.

Bynagari, N. B. (2016). Industrial Application of Internet of Things. Asia Pacific Journal of Energy and Environment, 3(2), 75-82. https://doi.org/10.18034/apjee.v3i2.576

Bynagari, N. B. (2017). Prediction of Human Population Responses to Toxic Compounds by a Collaborative Competition. Asian Journal of Humanity, Art and Literature, 4(2), 147-156. https://doi.org/10.18034/ajhal.v4i2.577

Bynagari, N. B. (2018). On the ChEMBL Platform, a Large-scale Evaluation of Machine Learning Algorithms for Drug Target Prediction. *Asian Journal of Applied Science and Engineering*, *7*, 53–64. Retrieved from https://upright.pub/index.php/ajase/article/view/31

Bynagari, N. B., & Fadziso, T. (2018). Theoretical Approaches of Machine Learning to Schizophrenia. *Engineering International*, *6*(2), 155-168. https://doi.org/10.18034/ei.v6i2.568

Cho, K., Merrienboer, B. van, Bahdanau, D., and Bengio, Y. (Oct. 7, 2014). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches.

Donepudi, P. K. (2014). Technology Growth in Shipping Industry: An Overview. American Journal of Trade and Policy, 1(3), 137-142. https://doi.org/10.18034/ajtp.v1i3.503

Donepudi, P. K. (2015). Crossing Point of Artificial Intelligence in Cybersecurity. American Journal of Trade and Policy, 2(3), 121-128. https://doi.org/10.18034/ajtp.v2i3.493

Donepudi, P. K. (2016). Influence of Cloud Computing in Business: Are They Robust?. Asian Journal of Applied Science and Engineering, 5(3), 193-196. Retrieved from https://journals.abc.us.org/index.php/ajase/article/view/1181

Donepudi, P. K. (2017). Machine Learning and Artificial Intelligence in Banking. Engineering International, 5(2), 83-86. https://doi.org/10.18034/ei.v5i2.490

Donepudi, P. K. (2018). Application of Artificial Intelligence in Automation Industry. *Asian Journal of Applied Science and Engineering*, *7*, 7–20. Retrieved from https://upright.pub/index.php/ajase/article/view/23

Douguet, D., Thoreau, E., and Grassy, G. (July 1, 2000). A Genetic Algorithm for the Automated Generation of Small Organic Molecules: Drug Design Using an Evolutionary Algorithm. In: J Comput-Aided Mol Des 14.5, pp. 449–466.

Elton, D. C., Boukouvalas, Z., Fuge, M. D., and Chung, P. W. (Aug. 5, 2019). Deep Learning for Molecular Design—a Review of the State of the Art. In: Mol Syst Des Eng 4.4, pp. 828–849.

Ertl, P. and Schuffenhauer, A. (June 10, 2009). Estimation of Synthetic Accessibility Score of Drug-like Molecules Based on Molecular Complexity and Fragment Contributions. In: J Cheminformatics 1.1, p. 8.

Fadziso, T., & Manavalan, M. (2017). Identical by Descent (IBD): Investigation of the Genetic Ties between Africans, Denisovans, and Neandertals. Asian Journal of Humanity, Art and Literature, 4(2), 157-170. https://doi.org/10.18034/ajhal.v4i2.582

Gao, W. and Coley, C. W. (Apr. 6, 2020). The Synthesizability of Molecules Proposed by Generative Models. In: J Chem Inf Model.

Gomez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernandez-Lobato, J. M., S ´ anchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. (Feb. 28, 2018). Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. In: ACS Central Sci 4.2, pp. 268–276.

Guimaraes, G. L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C., and Aspuru-Guzik, A. (May 30, 2017). Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models. arXiv: 1705.10843.

Hochreiter, S. and Schmidhuber, J. (Dec. 1, 1997). Long Short-Term Memory. In: Neural comput 9, pp. 1735–80.

Jensen, J. H. (Mar. 20, 2019). A Graph-Based Genetic Algorithm and Generative Model/Monte Carlo Tree Search for the Exploration of Chemical Space. In: Chem Sci 10.12, pp. 3567–3572.

Jin, W., Barzilay, R., and Jaakkola, T. (Feb. 12, 2018). Junction Tree Variational Autoencoder for Molecular Graph Generation. arXiv: 1802 . 04364.

Kadurin, A., Aliper, A., Kazennov, A., Mamoshina, P., Vanhaelen, Q., Khrabrov, K., and Zhavoronkov, A. (Dec. 22, 2016). The Cornucopia of Meaningful Leads: Applying Deep Adversarial Autoencoders for New Molecule Development in Oncology. In: Oncotarget 8.7, pp. 10883– 10890.

Kadurin, A., Nikolenko, S., Khrabrov, K., Aliper, A., and Zhavoronkov, A. (Sept. 5, 2017). druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico. In: Mol Pharm 14.9, pp. 3098–3104.

Krenn, M., Hase, F., Nigam, A., Friederich, P., and Aspuru-Guzik, A. (Mar. 4, 2020). Self-Referencing Embedded Strings (SELFIES): A 100% Robust Molecular String Representation. arXiv: 1905 . 13741

Kusner, M. J., Paige, B., and Hernandez-Lobato, J. M. (Mar. 6, 2017). Grammar Variational Autoencoder. arXiv: 1703.01925.

Landrum, G. (2006). RDKit: Open-Source Cheminformatics. URL: http://www.rdkit.org.

LeCun, Y., Bengio, Y., and Hinton, G. (May 2015). Deep Learning. In: Nature 521.7553, pp. 436–444.

Lehman, J., Clune, J., Misevic, D., Adami, C., Altenberg, L., Beaulieu, J., Bentley, P. J., Bernard, S., Beslon, G., Bryson, D. M., et al. (Nov. 21, 2019). The Surprising Creativity of Digital Evolution: A Collection of Anecdotes from the Evolutionary Computation and Artificial Life Research Communities. arXiv: 1803.03453.

Li, Y., Vinyals, O., Dyer, C., Pascanu, R., and Battaglia, P. (Mar. 8, 2018). Learning Deep Generative Models of Graphs. arXiv: 1803.03324.

Manavalan, M. (2016). Biclustering of Omics Data using Rectified Factor Networks. International Journal of Reciprocal Symmetry and Physical Sciences, 3, 1–10. Retrieved from https://upright.pub/index.php/ijrsps/article/view/40

Manavalan, M. (2018). Do Internals of Neural Networks Make Sense in the Context of Hydrology? . *Asian Journal of Applied Science and Engineering*, *7*, 75–84. Retrieved from https://upright.pub/index.php/ajase/article/view/41

Manavalan, M., & Bynagari, N. B. (2015). A Single Long Short-Term Memory Network can Predict Rainfall-Runoff at Multiple Timescales. International Journal of Reciprocal Symmetry and Physical Sciences, 2, 1–7. Retrieved from https://upright.pub/index.php/ijrsps/article/view/39

Manavalan, M., & Donepudi, P. K. (2016). A Sample-based Criterion for Unsupervised Learning of Complex Models beyond Maximum Likelihood and Density Estimation. ABC Journal of Advanced Research, 5(2), 123-130. https://doi.org/10.18034/abcjar.v5i2.581

Mayr, A., Klambauer, G., Unterthiner, T., Steijaert, M., Wegner, J. K., Ceulemans, H., Clevert, D.-A., and Hochreiter, S. (June 6, 2018). Large-Scale Comparison of Machine Learning Methods for Drug Target Prediction on ChEMBL. In: Chem Sci.

Merk, D., Friedrich, L., Grisoni, F., and Schneider, G. (Jan. 2018). De Novo Design of Bioactive Small Molecules by Artificial Intelligence. In: Mol Inform 37.1-2.

Merk, D., Grisoni, F., Friedrich, L., and Schneider, G. (Oct. 22, 2018). Tuning Artificial Intelligence on the de Novo Design of Natural-Product-Inspired Retinoid X Receptor Modulators. In: Nat Commun Chem 1.1, pp. 1–9

Neogy, T. K., & Bynagari, N. B. (2018). Gradient Descent is a Technique for Learning to Learn. *Asian Journal of Humanity, Art and Literature*, *5*(2), 145-156. https://doi.org/10.18034/ajhal.v5i2.578

Nguyen, A., Yosinski, J., and Clune, J. (Apr. 2, 2015). Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images. arXiv: 1412.1897.

O'Boyle, N. and Dalke, A. (Sept. 19, 2018). DeepSMILES: An Adaptation of SMILES for Use in Machine-Learning of Chemical Structures. chemrxiv: 7097960.v1.

Olivecrona, M., Blaschke, T., Engkvist, O., and Chen, H. (Sept. 4, 2017). Molecular De-Novo Design through Deep Reinforcement Learning. In: J Cheminformatics 9.1, p. 48.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-Learn: Machine Learning in Python. In: J Mach Learn Res 12.85, pp. 2825–2830

Polykovskiy, D., Zhebrak, A., Sanchez-Lengeling, B., Golovanov, S., Tatanov, O., Belyaev, S., Kurbanov, R., Artamonov, A., Aladinskiy, V., Veselov, M., et al. (Nov. 29, 2018). Molecular Sets (MOSES): A Benchmarking Plat form for Molecular Generation Models. arXiv: 1811.12823.

Popova, M., Isayev, O., and Tropsha, A. (July 1, 2018). Deep Reinforcement Learning for de Novo Drug Design. In: Sci Adv 4.7, eaap7885.

Preuer, K., Renz, P., Unterthiner, T., Hochreiter, S., and Klambauer, G. (Sept. 24, 2018). Frechet ChemNet Distance: A Metric for Generative Models for Molecules in Drug Discovery. In: J Chem Inf Model 58.9, pp. 1736–1741.

Rogers, D. and Hahn, M. (May 24, 2010). ExtendedConnectivity Fingerprints. In: J Chem Inf Model 50.5, pp. 742–754.

Sanchez-Lengeling, B. and Aspuru-Guzik, A. (July 27, 2018). Inverse Molecular Design Using Machine Learning: Generative Models for Matter Engineering. In: Science 361.6400, pp. 360–365.

Sanchez-Lengeling, B., Outeiral, C., Guimaraes, G. L., and Aspuru-Guzik, A. (Aug. 17, 2017). Optimizing Distributions over Molecular Space. An Objective Reinforced Generative Adversarial Network for InverseDesign Chemistry (ORGANIC). chemrxiv: 5309668. v3.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (Jan. 2009). The Graph Neural Network Model. In: IEEE Trans Neural Netw 20.1, pp. 61–80.

Schmidhuber, J. (Jan. 1, 2015). Deep Learning in Neural Networks: An Overview. In: Neural Netw 61, pp. 85–117.

Schneider, G. (2013). De Novo Molecular Design. John Wiley & Sons, Ltd.

Segler, M. H. S., Kogej, T., Tyrchan, C., and Waller, M. P. (Jan. 24, 2018). Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks. In: ACS Central Sci 4.1, pp. 120–131

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (Feb. 19, 2014). Intriguing Properties of Neural Networks. arXiv: 1312.6199.

Venkatasubramanian, V., Chan, K., and Caruthers, J. M. (Sept. 1, 1994). Computer-Aided Molecular Design Using Genetic Algorithms. In: Comput Chem Eng 18.9, pp. 833–844.

Weininger, D. (Feb. 1, 1988). “SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules”. In: J Chem Inf Comput Sci 28.1, pp. 31–36.

You, J., Liu, B., Ying, R., Pande, V., and Leskovec, J. (Feb. 24, 2019). Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. arXiv: 1806.02473.

Zhang, C., Lyu, X., Huang, Y., Tang, Z., and Liu, Z. (Nov. 18, 2019). Molecular Graph Generation with Deep Reinforced Multitask Network and Adversarial Imitation Learning. In: IEEE Int Conf Bioinform Biomed.

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. (Sept. 2019). Deep Learning Enables Rapid Identification of Potent DDR1 Kinase Inhibitors. In: Nat Biotechnol 37.9, pp. 1038–1040.

Zhou, Z., Kearnes, S., Li, L., Zare, R. N., and Riley, P. (Oct. 19, 2018). Optimization of Molecules via Deep Reinforcement Learning. arXiv: 1810.08678.

--0--