Direct Steering of *de novo* Molecular Generation using Descriptor Conditional Recurrent Neural Networks (cRNNs)

Panagiotis-Christos Kotsias[§], Josep Arús-Pous[§⊥], Hongming Chen[§], Ola Engkvist[§], Christian Tyrchan[¥], Esben Jannik Bjerrum[§*]

§ Hit Discovery, Discovery Sciences, Biopharmaceuticals R&D, AstraZeneca, Gothenburg, Sweden

¥ Medicinal Chemistry, Research and Early Development, Respiratory, Inflammation and Autoimmune (RIA), BioPharmaceuticals R&D, AstraZeneca, Gothenburg, Sweden

⊥ Department of Chemistry and Biochemistry, University of Bern, Freiestrasse 3, 3012 Bern, Switzerland

* Corresponding author: Esben.bjerrum@astrazeneca.com

Abstract

Deep learning has acquired considerable momentum over the past couple of years in the domain of de-novo drug design. Particularly, transfer and reinforcement learning have demonstrated the capability of steering the generative process towards chemical regions of interest. In this work, we propose a simple approach to the focused generative task by constructing a conditional recurrent neural network (cRNN). For this purpose, we aggregate selected molecular descriptors along with a QSAR-based bioactivity label and transform them into initial LSTM states before starting the generation of SMILES strings that are focused towards the aspired properties. We thus tackle the inverse QSAR problem directly by training on molecular descriptors, instead of iteratively optimizing around a set of candidate molecules. The trained cRNNs are able to generate molecules near multiple specified conditions, while maintaining an output that is more focused than traditional RNNs yet less focused than autoencoders. The method shows promise for applications in both scaffold hoping and ligand series generation, depending on whether the cRNN is trained on calculated scalar molecular properties or structural fingerprints. This also demonstrates that fingerprint-to-molecule decoding is feasible, leading to molecules that are similar – if not identical – to the ones the fingerprints originated

from. Additionally, the cRNN is able to generate a larger fraction of predicted active compounds against the DRD2 receptor when compared to an RNN trained with the transfer learning model.

Introduction

The disruptive impact deep generative models have delivered over the past couple of years in the domain of de-novo drug design is profound, due to offering the capability of directing the generative process towards chemical regions of interest [1–3]. Specifically, deep learning has found applications in biological tasks such as bioactivity and synthesis prediction, image processing and *de novo* design of novel molecules [4]. A challenging task deep networks tried to address is the inverse molecular design [5] which refers to the generation of molecular structures that meet desired conditions, such as specific physicochemical properties or properties predicted by Quantitative Structure-Activity Relationship (QSAR) models.

The Simplified Molecular-Input Line-Entry System (SMILES) [6] is a popular choice [7] to represent molecules when using recurrent neural networks (RNNs). The alphanumeric nature of SMILES strings makes them compatible with state-of-the-art natural language processing algorithms performing sequence modelling and generation, such as RNNs. In particular, RNNs are a widely accepted approach to the task of sequence modeling because of their ability to memorize past inputs and incorporate them into their inference [8].

Unbiased RNN generative models trained on a relatively small number of SMILES strings were shown to be able to cover a much larger chemical space [9]. Moreover, the augmentation of a dataset using SMILES with randomized atom order has demonstrated state-of-the-art performance with respect to the uniformity and completeness of the coverage of chemical regions, compared to simply using their canonical variants [10]. After learning the general rules of the chemical space, e.g. atom types, bond types and size of molecules, the prior network can be further specialized using smaller datasets in a transfer learning fashion [11] or using reinforcement learning [12–14].

More complicated architectures such as autoencoders [15], which include two jointly-trained neural networks responsible for converting the input to and back from a latent representation, have been extensively benchmarked [16, 17]. The quality of the latent space of an autoencoder was also proven to benefit from the usage of randomized SMILES strings [18–20]. Moreover, the latent space representation of a molecule can be used towards optimizing QSAR endpoints using GANs [21], Bayesian Optimization [15] or Particle Swarm Optimization [22]. The combination of a heteroencoder

[19], that is trained on pairs of randomized SMILES strings of the same molecule, with a generative adversarial network (GAN) [23] has further demonstrated automatic navigation towards properties of interest.

Alternatively, learning to precondition the structure generation eliminates the need for optimization loops. One approach demonstrated this capability by concatenating SMILES strings with the properties of interest as input to a variational autoencoder [24]. Molecular graphs [25] have also been used in pairs along with the desired change in properties as conditions to train a variational autoencoder on. Latent representations that are generated from a GAN architecture may also be exploited as input conditions for decoding neural networks [23].

In this work, we demonstrate that molecule side information, such as molecular descriptors, can be incorporated into the RNN-based generative process. We construct conditional recurrent neural networks (cRNNs) by setting the internal states of long short-term memory cells (LSTM [26]) as the input conditions. The architecture is related to, but conceptually simpler than, a conditional autoencoder as we only utilize an RNN-based decoder part. The generation is conditioned with properties calculated directly from the molecular structure or QSAR models, thus the encoder part is no longer needed. The conditional seed successfully steers the focus of the RNN towards a particular subset of the chemical domain, such as bioactive compounds with respect to a specific protein target. Our approach complements the existing state-of-the-art conditional generative models such as conditional VAE, reinforcement learning etc. and may be used for populating molecular libraries.

Methods

Datasets

The datasets used in this work originate from two publicly available sources: ChEMBL [27] and ExCAPE-DB [28]. Data from ChEMBL were used to train the generative neural network and data from the dopamine receptor D2 (DRD2) target data in ExCAPE-DB were used to train a Quantitative Structure-Activity Relationship (QSAR) model using a support vector classification model to estimate the potency of a generated compound towards DRD2.

ChEMBL

The neural network was trained with a subset of the ChEMBL version 25. Initially, the complete dataset has been standardized using the MolVS Python module [29] using the super parent setting, which standardizes fragment, charge, isotope, stereochemistry and tautomeric states. Molecules were filtered to only contain the atoms [H, C, N, O, F, S, Cl, Br] with total heavy atoms less than 50. Next, the known active molecules found in the DRD2 dataset (see below) were removed from the dataset. The dataset was split into training and test subsets with a 9:1 ratio. During training, 10% of the training subset was used as a fixed validation set.

ExCAPE-DB

All data regarding the DRD2 entry in ExCAPE-DB were downloaded [30] and preprocessed as follows: first, duplicate compounds as well as SMILES strings [6] that were not sanitizable by RDKit v2018.09.1 [31] were removed from the DRD2 dataset. All compounds with a pXC50 value greater than five were selected as known actives along with 100,000 random DRD2 measured inactive compounds from ExCAPEDB. Stereochemical information was removed by converting all molecules to non-isomeric SMILES strings. The dataset was further reduced to exclude SMILES strings that were longer than the ones in ChEMBL or contained characters not found in ChEMBL. This led to removing strings with iodine and phosphorus. All active molecules were clustered based on the pairwise Tanimoto distance of their Morgan fingerprints with a radius of two using the implementation of the Butina algorithm [32] found in RDKit. The maximum distance threshold for the algorithm to associate neighbours was fixed to 0.4 with a value above it dictating different clusters. All clusters were sorted based on their size and were assigned to the train, validation and test subsets iteratively using a “4-1-1” scheme, i.e. for every four clusters assigned to the train set, one cluster was assigned to the validation set and one cluster to the test set in order of decreasing cluster size.

SMILES strings Randomization and Vectorization

During training, the atom order of all molecules was randomized using RDKit. After converting them back to SMILES, every constituting character was one-hot encoded. Every SMILES string was thus represented by a two-dimensional array with dimensions corresponding to the length of the vocabulary and the maximum canonical SMILES length found in ChEMBL, with an offset of five extra characters to account for randomized SMILES which were longer than their canonical representation. The characters “^” and “$” were inserted in the beginning and end of each one-hot encoded string respectively. Resulting arrays that corresponded to shorter SMILES strings were padded with the end-character “$”. The considered vocabulary consisted of 35 tokens that included all common unique alphanumeric characters found in ChEMBL and DRD2 datasets after filtering, the delimiters “^” and “$” and the token “?” to account for one-hot encoding of unknown characters.

The randomization and vectorization of all SMILES strings was performed dynamically using a modified version of the molvecgen Python package [33] during training.

DRD2 QSAR Model

A probabilistic Support Vector Machine Classification model was used for bioactivity prediction. The standard implementation of a support vector machine from scikit-learn v0.20.3 [34] Python package was used, with the radial basis function as a kernel function. The model was trained to discriminate active compounds from inactive ones based on their 2048-bit radius 2 Morgan fingerprint representations. The hyperparameters C and γ were optimized with randomized search, where 50 different values per parameter were drawn from two exponential distributions with replacement.

Recurrent Neural Network

The neural network resembles the decoder architecture described in [19]. It was implemented in Keras v2.2.4 [35] with TensorFlowGPU v1.12.0 backend [36] and it is schematically shown in Figure 1. The network accepts a vector of molecular descriptors as inputs to a set of six Dense layers of 256 units each using the ReLU [37] activation function. The output of each individual Dense layer is used to set either the cell state or the hidden state of each of the recurrent layers of the network. There are in total three unidirectional recurrent layers in the network, each one consisting of 256 Long Short-Term Memory (LSTM) [26] neurons. The output of the final LSTM layer is fed to a feedforward layer with 35 units, which is the length of the character space, using softmax activation. Batch normalization was applied to the outputs of all LSTM and all but the last Dense layers. Keras CUDA-enabled CuDNNLSTM units were used in the recurrent layers.

The model was trained for 100 epochs with randomized SMILES strings following the Teacher’s Forcing method [38], using the ground truth at each step as prior knowledge instead of the character previously predicted by the network. A batch size of 128 sequences was used along with the Adam optimizer with default parameters [39] and an initial learning rate of 10^{-3}. A custom learning rate schedule was used, where the learning rate was kept constant for the first 50 epochs and then decayed exponentially at each epoch, down to a value of 10^{-6} at the final epoch.

A copy of the trained model was modified for the purpose of predicting single characters to jointly form SMILES strings. While maintaining the trained connection weights, the shape of the output of the last feedforward layer was set to a one-dimensional vector expressing the probability of sampling each of the known characters at every step. Also, the LSTM layers were set to stateful mode. During inference, a single character per iteration is sampled out of this vector of probabilities using multinomial sampling. After setting the initial states according to the descriptors of interest, the biased generation is triggered by feeding the start-character "^" to the network and ends when the end-character "$" is sampled.

Two different cRNN models were constructed and trained following this procedure, each based on different input descriptors. The first PhysChem-Based (PCB) model is shown schematically in Figure 1A. The model uses the Wildman-Crippen partition coefficient (LogP) [40], topological polar surface area (TPSA), molecular weight (MW), number of hydrogen bond acceptors (HBA), number of hydrogen bond donors (HBD) and the drug-likeness score (QED) [41] calculated using their RDKit implementations as well as the soft label predicted by the QSAR SVC model described above. The calculated values were scaled individually to achieve a distribution with zero mean and unit variance and they were concatenated. The second FingerPrint-Based (FPB, Figure 1B) model was trained solely on Morgan fingerprints of radius two and 2048 bits, which are similar to Extended Connectivity Fingerprints (ECFP). The training and inference schemes of the cRNN models are described in Figure 1A-B and Figure 1C respectively.

Model training and inferencing was performed on an NVIDIA Tesla V100 GPU on a 64-bit CentOS v7.5 server with 128 GB of RAM. The training process of the PCB and FPB models utilized 5 and 25 GB of RAM, respectively.

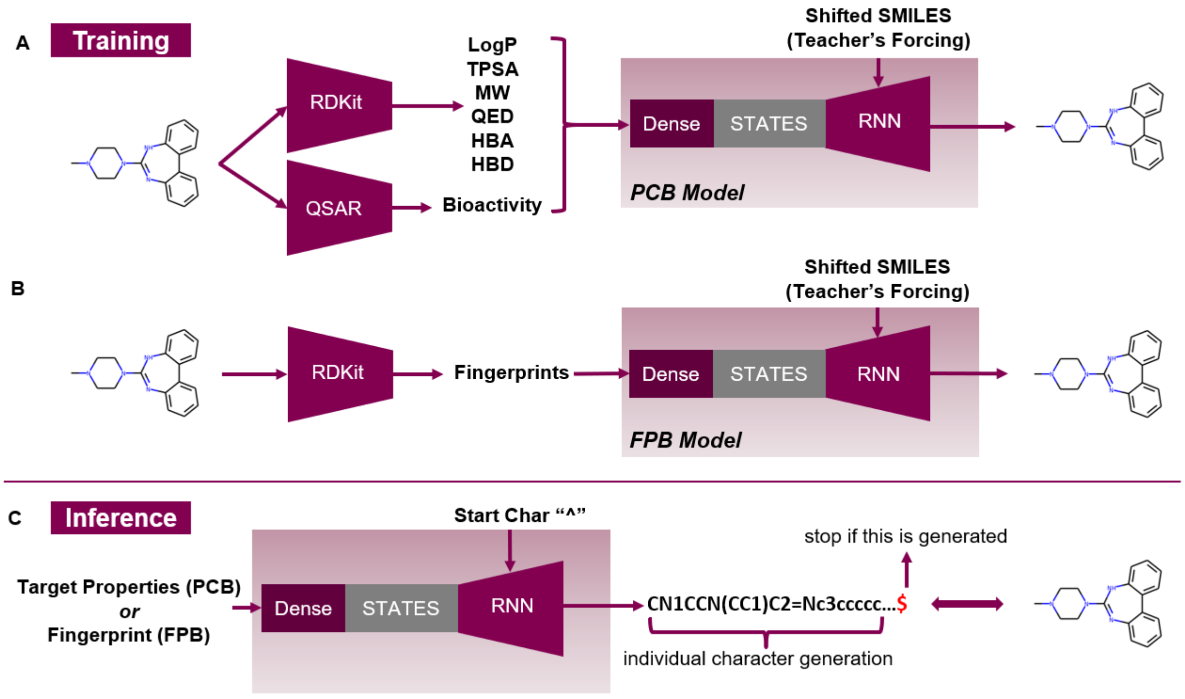
A
Training
RDKit
QSAR
LogP
TPSA
MW
QED
HBA
HBD
Bioactivity
Shifted SMILES (Teacher's Forcing)
Dense
STATES
RNN
PCB Model
B
RDKit
Fingerprints
Shifted SMILES (Teacher's Forcing)
Dense
STATES
RNN
FPB Model
C
Inference
Target Properties (PCB) or Fingerprint (FPB)
Start Char "^"
Dense
STATES
RNN
CN1CCN(CC1)C2=Nc3ccccc...$
stop if this is generated
individual character generation

Figure 1: Conditional Recurrent Neural Network (cRNN) models based on different conditions. ***A)*** *The PhysChem-Based (PCB) model accepts six scalar properties calculated by RDKit Python library and concatenates them with the probabilistic bioactivity prediction of the QSAR model.* ***B)*** *The FingerPrint-Based (FPB) model accepts a 2048-bit Morgan fingerprint vector calculated by RDKit. Both models are trained on randomized SMILES strings as targets. C) Model inference is biased by the conditional seed and triggered by the starting character "^". Inferencing stops when "$" is generated.*

Transfer Learning Model

The baseline model consists of the same neural network architecture as described above with the notable difference that the initial states, instead of being set based on known descriptors, are rather being reset to zero in the beginning of the generation of each string. This approach is similar to the prior network described in [12] with the difference that each character is treated independently rather than within multi-character tokens. The network was likewise trained with Teacher's Forcing, learning the character set and the grammar of the SMILES strings found in ChEMBL. The selected RNN dimensions were identical to the ones in the case of the conditional RNN.

Next, the prior model was further trained exclusively with the known actives of the DRD2 train dataset for an additional 200 epochs, following a transfer learning strategy [42]. The initial learning rate was set to $10^{-4.5}$ and it was decayed exponentially down to 10^{-6} by the end of the training.

Likelihood of Known Sequences

The likelihood of sampling a given SMILES strings was estimated using the negative log likelihood (NLL) as previously described [9], with a modification that incorporates the knowledge that is induced into the initial states of the generation in the case of a conditional model. The conditional negative log-likelihood is described in Equation 1.

$$CNLL(S|c) = -\left[\ln P(X_1 = T_1 \mid c) + \sum_{i=2}^{N} \ln P(X_i = T_i|\ X_{i-1} = T_{i-1}, \dots, X_1 = T_1, c)\right]$$

(Equation 1)

T_i are the characters in the known SMILES sequence *S*, X_i are the predicted model outputs, *N* is the length of the sequence *S* and *c* refers to the seeding conditions. The sign of the log-likelihood is negated to reflect that higher values correspond to more improbable sequences.

Results and Discussion

Resulting Datasets

The filtering process described in the previous section resulted in the sizes of datasets shown in Table 1. The QSAR Support Vector Classification model with parameters C=5.53 and γ=0.022 was selected as the one with the highest F1-score (0.92) towards the DRD2 validation set. This model was used to label all compounds in the ChEMBL dataset leading to 2.3% of ChEMBL compounds being classified with a probability greater than 50% of being active against the DRD2 receptor (Table 1). As shown in Figure 2, the property distribution of the two datasets largely follows each other, except that the QED score of the DRD2 dataset is shifted towards higher values since those molecules are expected to be a priori drug-like.

Table 1: Size and percentage of active compounds per dataset

Dataset	Total Samples	Active %
DRD2_TRAIN	71,512	6.7(1)
DRD2_VALID	17,800	6.3(1)
DRD2_TEST	17,817	6.4(1)
CHEMBL_TRAIN	1,347,173	2.3(2)
CHEMBL_TEST	149,679	2.3(2)

(1): Known active compounds
(2): Predicted active compounds (probability ≥ 0.5) by the QSAR model

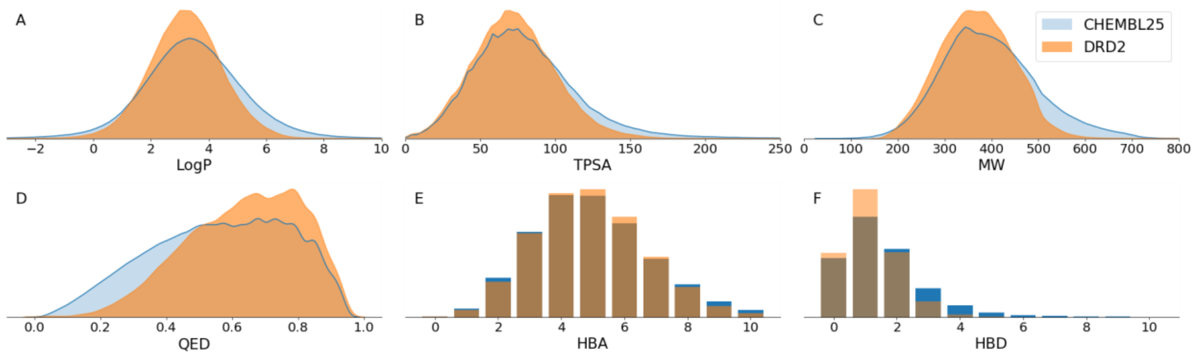

*Figure 2: Distribution of **A)** lipophilicity (logP), **B)** topological polar surface area (TPSA), **C)** molecular weight (MW), **D)** drug-likeness score (QED), **E)** number of hydrogen bond acceptors (HBA) and **F)** hydrogen bond donors (HBD) with respect to the complete CHEMBL25 and DRD2 datasets before splitting. Subfigures A-D show the continuous histogram density as estimated by the kdeplot method of the seaborn Python library using default parameters.*

NLL distributions of datasets

The NLL of sampling all molecules in the ChEMBL25 dataset and all known active compounds in the DRD2 dataset was calculated with respect to all different models based on canonical SMILES strings. Figure 3 shows all different NLL distributions using the smoothened estimate of the density function of the underlying histograms.

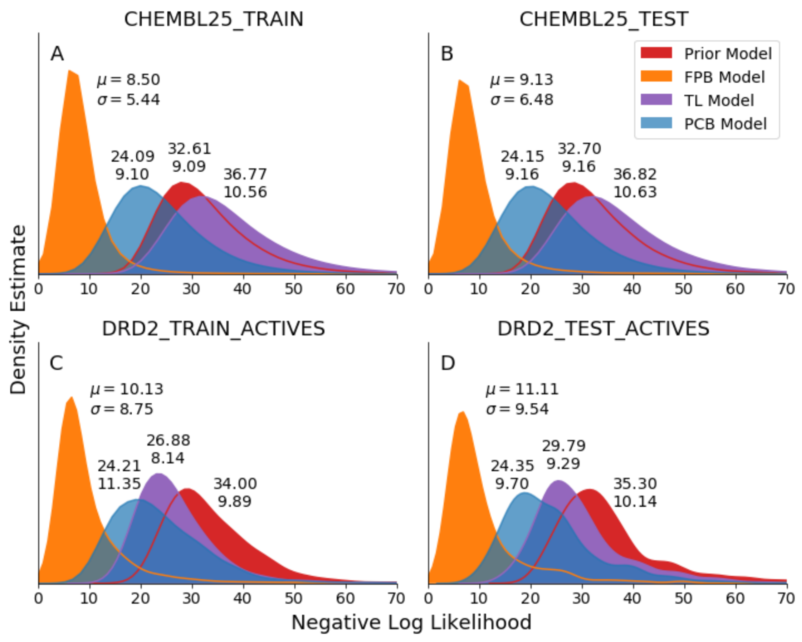

Figure 3: Negative Log Likelihood (NLL) of sampling all canonical SMILES per dataset with the physchem-based (PCB), fingerprint-based (FPB), transfer learning (TL) and prior models. The plots show the estimate of the density of the underlying NLL histograms. The mean and standard deviation of the true distributions are annotated. The CHEMBL25 sets consist of both predicted active and inactive compounds whereas only the known actives were selected from the DRD2 sets for this test. The graphs are truncated at a maximum NLL value of 70.

In all datasets the FPB model results in the sharpest distribution of NLL with the lowest mean and variance compared to the other three models. Likewise, the PCB model shows the second lowest NLL mean value per dataset. The order of the distributions plots is expected because the amount of chemical information in the 2048 bits of a Morgan fingerprint exceeds the information that is contained within the seven scalar descriptors used in the PCB model, especially from a structural point of view. The graphs of the conditional models show a slight shift towards higher values for the DRD2 datasets, due to the uncertainty that is inherent to unseen data. Nevertheless, both conditional models have a lower mean NLL – and thus a higher probability – of sampling the canonical SMILES the conditions originated from, compared to the prior network both before and after being trained with transfer learning. The transfer learning model curve changes its relative position compared to the prior model curve between the two datasets because the focus of the model trained with transfer learning has been shifted away from the majority of molecules in ChEMBL and, thus it is more difficult to sample them.

Ideally, all models should be able to sample the intended chemical space uniformly and this would be expressed by zero variance of the NLL distribution and the NLL curve should approximate a Dirac distribution. Under such ideal conditions, it would be possible to estimate the size of the output space by simply inverting the (constant) probability of sampling any molecule, e.g. a probability of 0.01 would mean that in total 100 molecules could be sampled. As an example, a sharp NLL distribution around a value of 10 would imply a uniform probability distribution at a value of $4.54 \cdot 10^{-5}$ or an equiprobable output space of 22,000 unique randomized SMILES strings. Likewise, NLL values of 20 and 30 would point to output domains of approximately 10^{8} and 10^{13} SMILES strings respectively. Those numbers would serve as an upper boundary for underlying unique molecules because the same molecule may be represented by multiple randomized SMILES. Even though the distributions of Figure 3 are far from Dirac distributions, a comparison of the distributions may serve as a qualitative insight on the relative change in the order of magnitude of their output space.

Additionally, the position of the distributions can be interpreted in two ways: first, the closer to zero the NLL distribution moves, the more deterministic the output of the model gets. This can be due to either limited generalizability of the model or more detailed description of the target, such as in the case of a conditional network. Second, differences in NLL distributions between train and test sets can be a sign of overfitting or mode collapse [9]. This seems to be the case with the transfer learning model, which exhibits a distribution with a lower mean NLL towards the active compounds in the DRD2 train dataset compared to the unseen active ones in the DRD2 test set. In contrast, the NLL distributions of sampling all four datasets with either of the conditional networks regardless of the dataset are on par, which makes overfitting a less likely cause. Here, the similar distributions regardless of a dataset demonstrates that the conditional models are able to generate both active and inactive compounds at equal ease, given that the states are set accordingly.

Sampling of Active Molecules

The structures shown in Figure 4 were generated by the two conditional networks using known active compounds from the DRD2 test set as conditional seeds, which are shown in the centre. The exemplified molecules in the dashed circle were generated by the FPB model whereas the ones outside of it were generated by the PCB model. All the molecules were filtered to have a QED score greater than 0.8 and were predicted to be active by the QSAR model with a probability greater than 0.8.

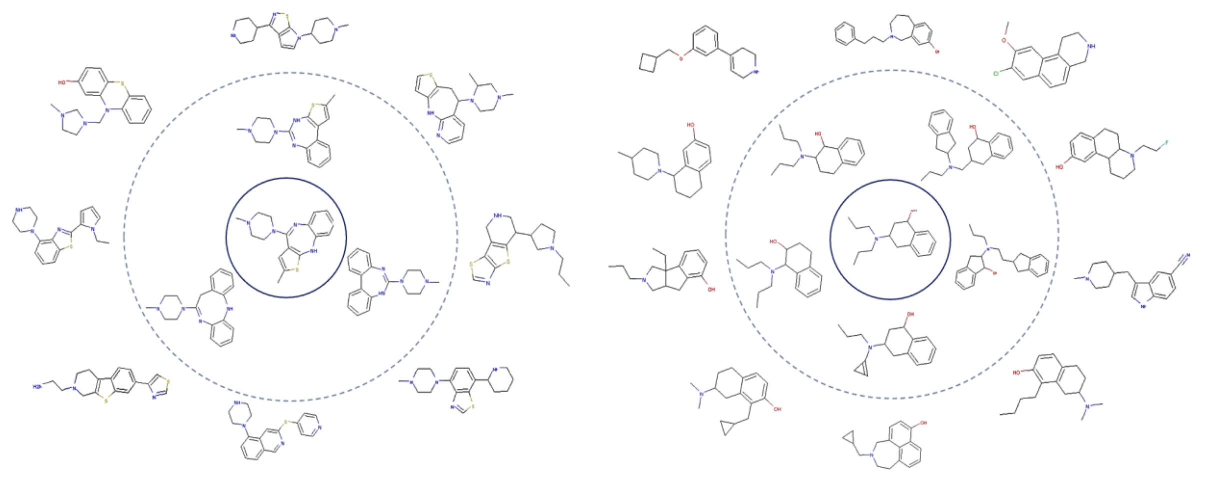

Figure 4: Generated structures from two different known active seeds (center) selected randomly from the DRD2 test set. The FPB (within dashed circle) and PCB (outside of dashed circle) generations have QED ≥ 0.8 and a predicted active probability ≥ 0.8. The FPB-generated molecules mostly maintain the seeding scaffold whereas the PCB-generated ones hop scaffolds.

The FPB-based generations demonstrate almost identical structure to the seed at least at a scaffold level. On the other hand, the PCB-generated molecules have clearly different scaffolds to seed, which can be attributed to the fact that the selected physicochemical descriptors do not encode structural information directly.

The correlation between the seed and the output of the models was further investigated by calculating the Tanimoto similarity of multiple generations. For that purpose, 100 seeds were randomly selected from the unseen active compounds of the DRD2 test set and for each one, 256 molecules were generated in a batch by each of the conditional models yielding a total of 25,600 SMILES strings. For each batch, the pairwise Tanimoto similarities were calculated between the scaffolds of the associated seed and of all uniquely generated compounds. The results are plotted in Figure 5A, while the predicted probabilities of being active for each compound were plotted in Figure 5B. The PCB-generated scaffolds tend to be dissimilar to their seeds in contrast to the FPB-generated ones, the similarity of which to the seeding scaffolds follows a bimodal distribution that is shifted to the right, showing that similar or identical scaffolds are generated. However, in both cases the distribution of active probabilities is comparable (Figure 5B), proving that both models can generate predicted active compounds given the appropriate conditions.

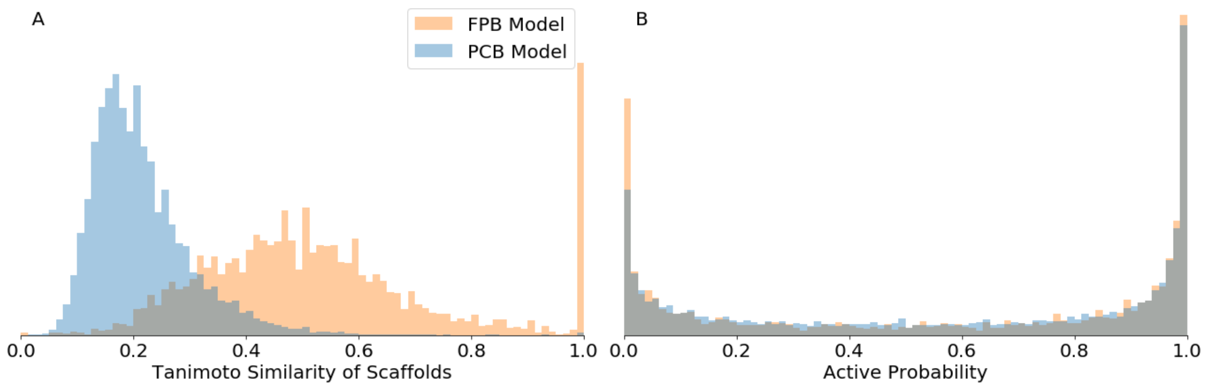

Figure 5: Distribution of pairwise Tanimoto similarity of scaffolds with the seed and predicted active probability of all unique generated structures per model. A) The PCB model generates new scaffolds whereas the FPB model generates structures that are more similar or even identical to the seed. B) Both models generate compounds that are predicted to be active with similar probability distributions.

This supports the previous observation that the PCB model can generate different scaffolds from the same seed. Additionally, it identifies the sampling domain of each model. The main advantage of using fingerprints is that structural restrictions are directly encoded, a fact that is of use when scaffolds that are similar or identical to the seed need to be generated. On the other hand, using physicochemical properties as conditions offers a more versatile sampling and this model could thus be applicable to explorations outside of a known scaffold, yet within the boundaries of the desired property setpoints.

Benchmarking

In an attempt to further compare all models, all three of them were tested with respect to the metrics provided by the MOSES framework [16]. For that purpose, 25,600 compounds were additionally sampled by the model trained with transfer learning, similar to the sampling done for the other models as described above. The metrics were calculated with respect to the active compounds of the DRD2 test set that was used as a reference dataset.

Table 2: Comparison of the physchem-based (PCB), fingerprint-based (FPB) and transfer learning models (TL) with respect to MOSES and custom metrics. The DRD2 test set was used as a reference set for the MOSES framework such that the seed conditions were drawn from it. Top-pointing arrows show that higher scores are considered better and bottom-pointing arrows show that lower scores are considered better. Numbers in bold show the best score for that metric. Molecules with a predicted probability greater than 0.5 by the QSAR model were considered active.

	Metrics		**Models**		
			PCB	*FPB*	*TL*
MOSES	Valid	↑	0.881	0.951	**0.968**
	Unique@1k	↑	0.996	0.276	**1.000**
	Unique@10k	↑	**0.996**	0.304	**0.996**
	FCD	↓	7.981	**5.590**	8.438
	SNN	↑	0.341	**0.774**	0.375
	Frag	↑	0.920	**0.966**	0.938
	Scaf	↑	0.094	**0.491**	0.193
	IntDiv	↑	0.845	0.834	**0.846**
CUSTOM	Novelty (1)	↑	0.878	0.299	**0.953**
	Predicted Active Fraction (2)	↑	**0.536**	0.194	0.474
	Reconstructability (3)	-	≤0.001	0.630	-

(1): calculated with respect to the generated **active** compounds and the merged **active** compounds of the DRD2_TRAIN and CHEMBL25_TRAIN datasets only
(2): fraction of 25,600 generations that are **valid**, **unique** and predicted **active** compounds
(3): calculated based on the most frequently sampled SMILES string out of 256 generations per conditional seed

The PCB model performs the worst with respect to most metrics, except for predicted active fraction and uniqueness among 10,000 samples. However, the metrics need to be interpreted carefully. The seed conditions used for the generation were extracted from active compounds of the DRD2 test set, which were not included in the training set of both conditional models. The active class is heavily underrepresented in the datasets that they were trained on (only 2.3% of predicted actives in ChEMBL, cf. Table 1) and thus the set of conditional seeds correspond to a demanding task, which becomes even harder for the PCB model to fulfil since much less information is included in the physicochemical descriptors than in the fingerprints. On the contrary, the transfer learning model was trained directly on known actives and it is independent of any input during generation, while trying to replicate what has been seen during training. Lacking input conditions offers an implicit advantage over the conditional models in terms of valid generated structures, because specific input combinations may cause a consistent drop in generated validity. However, within a sample of 10,000 generated structures, the PCB and the transfer learning models are on par regarding uniqueness. This metric is a performance indicator, yet it does not fully expose the differences between the models, simply because the output space is too large to generate enough duplicates within only 10,000 samples. On the other hand, the FCB model has low uniqueness, but this is expected as the more deterministic nature and the lower number of possible SMILES to sample from a single fingerprint naturally lead to duplicated outputs and penalized uniqueness.

The Frechét ChemNet distance (FCD) [43] underlines the chemical distance between the reference and the generated distributions. As such, it is heavily in favour of the conditional models, because the seeds drawn from the test set purposely force the generated distributions towards it and consequently towards lower FCD values. Moreover, since the DRD2 train and test sets had been clustered, the fact that the transfer learning model was further trained on one of them explains the deviation from the other with respect to the FCD metric. Internal diversity exhibits also expected behaviour for a similar reason; the seeds narrow the output down compared to an ideally random sampler in the DRD2 active domain, such as the transfer learning model.

Among all 25,600 molecule generations, higher novelty was achieved by the transfer learning model. This was due to the lower validity and uniqueness of the conditional models because the upper novelty boundary is defined by the product of validity and uniqueness. For the PCB model that boundary is approximately $0.881 \cdot 0.996 = 0.877$ which is reached as seen in Table 2. This is an indicator that the PCB model, even though it suffers from lower validity compared to the transfer learning model, it does not copy the training dataset. Similarly, the upper novelty boundary for the FPB is around $0.951 \cdot 0.304 = 0.289$ which was slightly exceeded because uniqueness@10k calculated by MOSES is probably lower than the complete uniqueness of all 25,600 generations. For the transfer learning model, the upper boundary $0.968 \cdot 0.996 = 0.964$ was almost reached as well. As observed, even though the absolute number of novel compounds was higher for the transfer learning model, none of the models actually replicated the training datasets.

It is noteworthy that a higher fraction of unique predicted active compounds was sampled by the PCB model whereas the least of them were generated by the FPB model. The FPB model was punished because of its high reconstructability, which negatively affects its uniqueness score.

More specifically, the molecular reconstructability of the input descriptors was assessed by trying to retrieve the molecule that was represented by them at each batch. By identifying the most frequently sampled molecule out of 256 generations using a single conditional seed, almost 65% of the FPB generations were proven to be successful reconstructions of the molecule behind the seeding fingerprint. Further experimentation with a deeper FPB model with four decoding layers and 512 LSTM units each made it possible to increase the reconstructability to 72%. Nonetheless, reconstructions were very scarce when using the physicochemical descriptors in the PCB model, because 256 samples were not enough to completely rediscover the diverse molecular space behind a given input condition.

In order to investigate whether novelty of the conditional models is influenced by the training or seeding dataset, 100 new conditions were drawn from each one of the training and test subsets of ChEMBL. Then, the novelty of the unique valid generated structures out of 256 generations (one batch) per set of conditions was assessed with respect to both datasets. The results are shown in Figure 6. As hypothesized, both models use the conditions stemming from unseen molecules and generate structures that are not present in either dataset. For any of the models, the difference between datasets is insignificant, reflecting a consistent generation of novel compounds regardless of the origin of the seeding conditions.

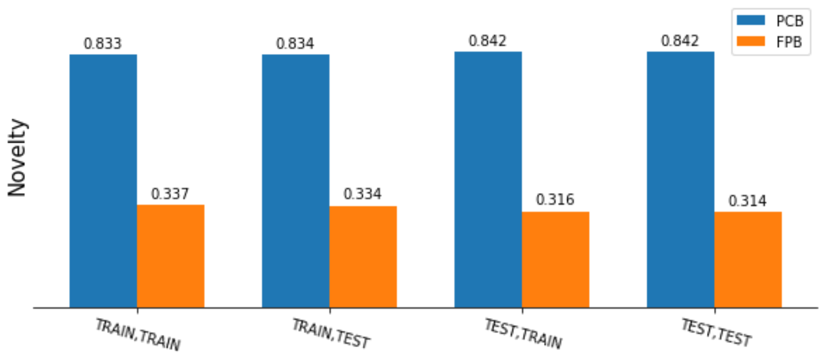

Figure 6. Novelty of generated molecules with respect to the train and test ChEMBL datasets using the physchem-based (PCB) and fingerprint-based (FPB) models. The first element of every pair on the x-axis corresponds to the dataset the conditions were drawn from. The second element represents the dataset with respect to which novelty was calculated. For any model the difference between datasets is insignificant, reflecting a consistent generation of novel compounds regardless of the seeding conditions. The numbers correspond to the fraction of valid unique novel molecules out of 25,600 generations.

Control of Generated Properties

The primary advantage of the PCB model is the ability to generate molecules that follow the desirable properties. This was tested by using 10 conditional seeds derived from randomly selected active compounds from the DRD2 test set whose QED scores are all greater than 0.8. For each conditional seed, a batch of 256 SMILES were generated and the physicochemical properties defined in the condition were calculated for all the generated valid molecules using RDKit. As shown in Figure 7, most of the properties of generated compounds exhibit only small deviation from the defined conditional setpoint, with the QED property having relatively large variance around the reference level.

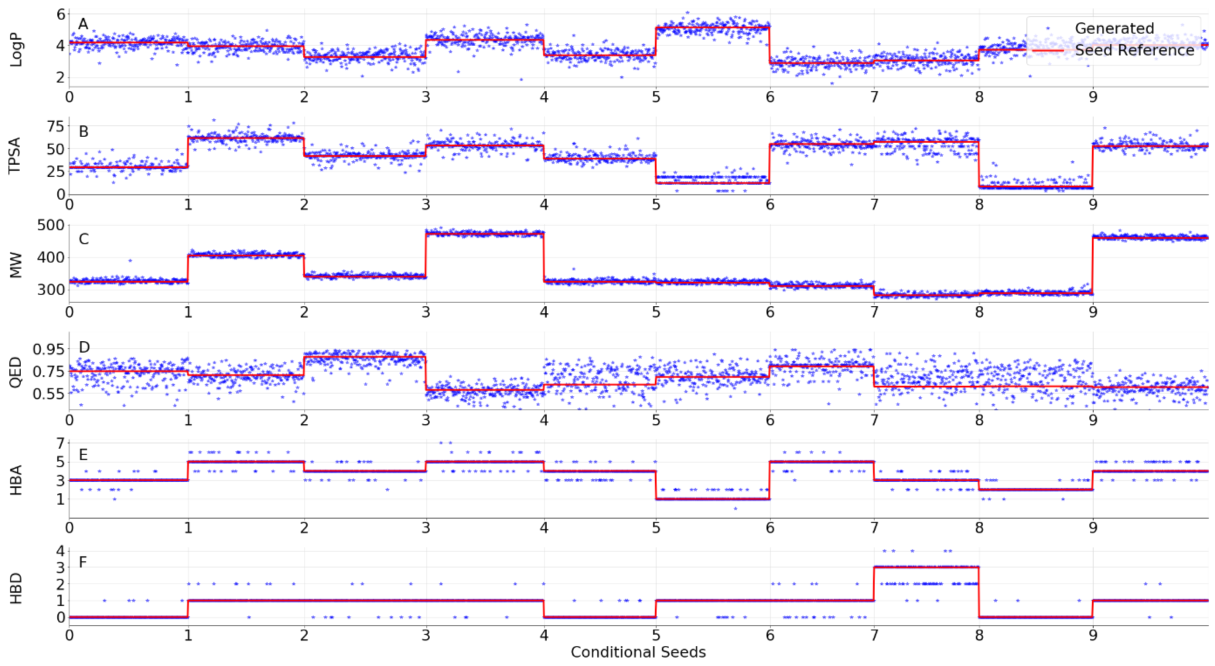

Figure 7: Reference properties (red) vs. properties of generated compounds (blue) for 10 random conditional seeds from the DRD2 test set. The length of the step denotes the amount of valid SMILES generations out of a batch of 256. The patterns of all generated properties follow the reference. The QED constraint is the hardest to satisfy.

To further investigate the capability of a cRNN to control the properties of its generated molecules, more experiments were conducted where single properties were varied in both directions while keeping the rest of them fixed. A molecule from within the first and third quartiles with respect to all properties of the DRD2 test dataset was selected to obtain the initial conditions from. Then, for each of the descriptors apart from the active probability, five values were tried out in a step-wise ascending fashion spanning the value range between the first and third quartile of each property, while keeping the rest of the conditions at the initial level. The tested conditions correspond to arbitrary property setpoints, unlike the ones shown in Figure 7. The reference (red line) and generated properties (blue dots) of all valid SMILES strings are shown in Figure 8. Each column of cells per plot corresponds to the tuning of a single property while keeping the other five conditions fixed at the initial values. In overall, LogP, TPSA, molecular weight and HBD setpoints were adequately matched in the generated molecular properties, followed by HBA which seems to be unstable for low values of LogP and high values of molecular weight. The QED formula contains the weighted sum [41] of all the other five properties and, consequently, the requested conditions along with the QED setpoint may render it impossible for that equation to be satisfied. Therefore, the QED property was hard to keep at the reference value as shown by the large spread around the target value (Figure 8).

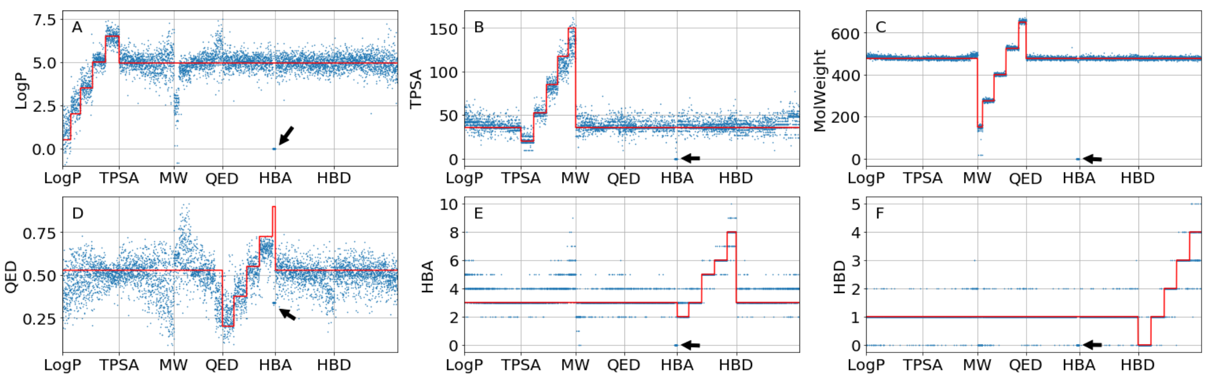

Figure 8: Optimization of conditions varied individually in every direction. The pattern of the properties of the generated molecules (blue dots) seems to follow the set conditions (red lines). The length of a step represents the amount of valid generations for that setpoint out of 256 samples. Low molecular weight or high QED setpoints lead to unstable generation of valid SMILES for the given condition. QED displays the largest deviations from the seed conditions and is the hardest property to control as the formula contains a weighted sum of the other 5 properties [41]*. The area annotated by arrows refers to an input combination with a high QED target that caused the generations to collapse with respect to the rate of valid SMILES and fulfillment of the specified conditions.*

This is particularly evident in the region around low values of molecular weight or requested high values of QED for the given seed. In the first case, LogP and QED decreased under the influence of the value of molecular weight that was controlled by the cRNN. From the short length of the step, it is observed that this batch of generations suffered from a high number of invalid SMILES strings and as far as the valid ones are concerned, their LogP and QED were much lower than the setpoint. Similarly, requested high values of QED given the values of the other 5 properties were impossible to achieve, which affected the values of all properties and eventually led to none of their setpoints being respected, as annotated with the arrow markings in Figure 8.

Those are cases in which input combinations were ill-defined and resulted to either unattractive molecules or invalid structures, something that has also been observed in the latent space vectors of autoencoders [21]. In the cRNN context, such combinations may refer to under-represented regions in the training dataset, which are either due to the lack of relevant samples in the source or due to conflicts between the requested descriptor ranges. The conditions are entangled since they depend on each other, as observed from the behaviour of the QED score. In most cases, the user is probably interested in tuning only one of the properties rather than restraining many of them; nonetheless, the property conditions ought to be set at reasonable values to avoid the entanglement problem. More sophisticated sampling approaches, such as the LatentGAN architecture [23], could potentially address the entanglement problem. Particularly, the generator component of the LatentGAN may be used to autonomously propose a valid combination of input properties that lead to active generations towards bioactivity targets.

Exclusivity of Sampling

Sampling the cRNN model with the seed conditions derived from a query structure should theoretically make it more likely to generate structures similar to the seed (c.f. Figure 3) and less likely to sample dissimilar molecules. To investigate this hypothesis, 100,000 molecules were randomly selected from the ChEMBL test set and clustered using the DBSCAN algorithm [44], based on the Euclidean distance of their five scaled physicochemical properties (LogP, TPSA, MW, HBA, HBD). A value of ε = 0.1 and 10 minimum samples for associating core points were selected as parameters of the DBSCAN algorithm.

Next, two clusters of molecules (with size of 53 and 57 respectively) were manually selected to keep the variance of their descriptors within a range as narrow as possible, with preferably small overlap. The distributions of LogP, TPSA and MW of the selected clusters are shown in Figure 9A-C. All the molecules of the first cluster resulted in an HBA count of four and an HBD count of zero and all molecules of the second cluster resulted in counts of four and one respectively. The selected clusters show minimal or no overlapping with respect to LogP, TPSA and HBD count whereas they share the same count of HBA and similar values of MW. The values of QED and predicted probability of being active were not considered during clustering.

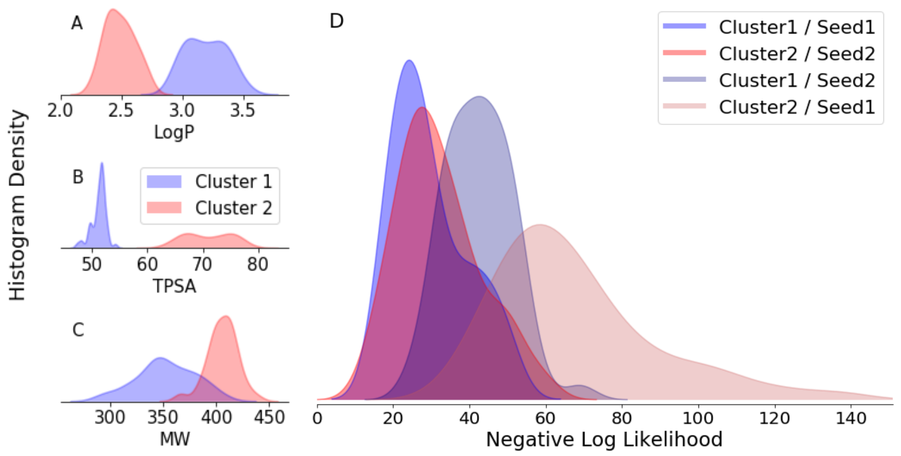

*Figure 9: **A-C)** Distribution of Properties of each cluster. **D)** Distribution of calculated Negative Log Likelihood of sampling each cluster using the two cluster centers as seeds interchangeably. It is shown that using a relevant seed makes it more probable to sample chemically neighboring molecules than molecules from another cluster.*

The seed conditions were selected as the coordinates of the geometric centre of each cluster. The conditional NLL of sampling the canonical SMILES of each cluster under different seeds was calculated according to Equation 1 (Figure 9D). The cross-conditional NLL was calculated for each cluster by swapping the conditional seeds of both clusters. Theoretically, the generation of molecules from these two clusters during conditional sampling should be mutually exclusive using their own cluster centre as seed. In other words, using each cluster centre as the seed should have a higher probability to sample the compounds within the same cluster.

This hypothesis is actually supported by Figure 10D. In overall, the molecules in cluster 1 (blue curve) are more likely to be sampled than the ones in cluster 2 (green curve), when the conditional vector is derived from the seed of cluster 1. When the seeds of both clusters are swapped, it is less probable to sample any molecule from one cluster using the seed of the other. Even though there is an overlap between the self-conditional and cross-conditional NLL curves, in both cases the former ones describe lower NLL values, thus showing that relevant molecules are more likely to be sampled. By comparing the self-conditional NLL curves of Figure 9 to the curves of Figure 3, it is observed that the NLL curves of Figure 8 are all shifted towards higher NLL values. This is expected though, since the conditions considered for each SMILES string were not derived from its own properties but from the mean properties of the cluster instead.

Applications to Drug Discovery

The cRNN architecture provides a way to address the inverse QSAR problem directly. Particularly, the PCB cRNN is able to generate molecular structures with desired properties. In contrast, other available methods suggest the use of optimization algorithms [15][22] or reinforcement learning [12] to close the loop and steer one or more initial candidate molecules towards the aspired region of the chemical domain in an iterative process. Such optimization approaches require computationally expensive looping over a cost or desirability function, whereas in our case a batch of 256 potentially interesting SMILES strings with properties close to predefined target values can be generated within less than a second. Runtime is of concern when scaling up the generative process for the design of diverse molecular libraries, the populating of which in a timely and interactive fashion might be essential. Having a quasi-instant generation also allows interactive applications to be built, where a fast feedback cycle permits experimentation with the target properties for library generation.

The pretrained cRNN could also serve as a starting point for additional optimization techniques such as reinforcement learning [12], where the standard conditions combined with the cRNN are complemented with project specific QSAR and desirability functions. Other optimization techniques such as Particle Swarm Optimization [22] or Bayesian Optimization [15] could be used to optimize the conditional seed, which in the case of the PCB model would be directly interpretable and for the FBP model could yield a series of similar compounds.

Conclusions

In this work, the effect of introducing molecular descriptors as inputs to an existing SMILES generator architecture based on recurrent neural networks has been investigated. Primarily, it was shown that known molecules are more likely to be rediscovered when sampling using the descriptor conditions that represent them as inputs to a cRNN, compared to a prior unbiased model that is simply trained on the complete molecular dataset. Our approach also demonstrated the capacity of generating novel compounds that were predicted active against the DRD2 receptor, which were also chemically closer to known active compounds than a baseline model trained with transfer learning. Additionally, a larger fraction of predicted actives was generated by the cRNN than the baseline model. Using molecular fingerprints as conditions focuses the molecular generation even more than physicochemical properties, by acting as structural restrictions that impose a scaffold on the output that is similar, if not identical, to the reference. This also demonstrated the proposed architecture's capability to function as a fingerprint inverter, by being able to resample the original molecule even up to 72% of the time by using a more complex network. On the other hand, physicochemical properties are more versatile and lead to molecules with more diverse structures and different scaffolds than the molecule that the conditions were derived from. The cRNN architecture tackles the inverse QSAR problem by directly shaping the properties of the generated molecules while avoiding computationally expensive optimization loops. Nonetheless, even though we have been able to optimize the conditions independently of each other, not all input combinations led to valid structures due to the conditions being correlated. As an example, this was observed when conditioning with a high QED setpoint while keeping the other conditions, which are constituents of the QED score calculation, fixed. The cRNN has thus been demonstrated as a potentially useful architecture with intermediate output space between unbiased character-based RNNs and fully steered autoencoders with a 1:1 relation between latent space vectors and molecules.

Acknowledgments

The authors would like to thank the whole of MolecularAI team at AstraZeneca for their invaluable input and the fruitful discussions held during the development of the present work.

Availability

The Python code and the trained neural networks used in this work can be found in the following Deep Drug Coder (DDC) GitHub repository: **https://github.com/pcko1/Deep-Drug-Coder**, that also includes an optional encoding network to constitute a molecular heteroencoder.

Conflicts of Interest

None to declare.

References

1. Xu Y, Lin K, Wang S, et al (2019) Deep learning for molecular generation. Future Med Chem 11:567–597. https://doi.org/10.4155/fmc-2018-0358

2. Elton DC, Boukouvalas Z, Fuge MD, Chung PW (2019) Deep learning for molecular design—a review of the state of the art. Mol Syst Des Eng. https://doi.org/10.1039/c9me00039a

3. Vamathevan J, Clark D, Czodrowski P, et al (2019) Applications of machine learning in drug discovery and development. Nat Rev Drug Discov 18:463–477. https://doi.org/10.1038/s41573-019-0024-5

4. Chen H, Engkvist O, Wang Y, et al (2018) The rise of deep learning in drug discovery. Drug Discov Today 23:1241–1250. https://doi.org/10.1016/j.drudis.2018.01.039

5. Sanchez-Lengeling B, Aspuru-Guzik A (2018) Inverse molecular design using machine learning: Generative models for matter engineering. Science (80-) 361:17. https://doi.org/10.1126/science.aat2663

6. Weininger D (1988) SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules. J Chem Inf Comput Sci 28:31–36. https://doi.org/10.1021/ci00057a005

7. Schwalbe-Koda D, Gómez-Bombarelli R (2019) Generative Models for Automatic Chemical Design. arXiv:190701632 1–25

8. Lipton ZC, Berkowitz J, Elkan C (2015) A Critical Review of Recurrent Neural Networks for Sequence Learning. arXiv:150600019 1–38

9. Arús-Pous J, Blaschke T, Ulander S, et al (2019) Exploring the GDB-13 chemical space using deep generative models. J Cheminform 11:0–34. https://doi.org/10.1186/s13321-019-0341-z

10. Arús-Pous J, Johansson S, Prykhodko O, Bjerrum EJ (2019) Randomized SMILES strings improve the quality of molecular generative models. ChemRxiv

11. Segler MHS, Kogej T, Tyrchan C, Waller MP (2018) Generating focused molecule

libraries for drug discovery with recurrent neural networks. ACS Cent Sci 4:120–131. https://doi.org/10.1021/acscentsci.7b00512

12. Olivecrona M, Blaschke T, Engkvist O, Chen H (2017) Molecular de-novo design through deep reinforcement learning. J Cheminform 9:. https://doi.org/10.1186/s13321-017-0235-x

13. Zhou Z, Kearnes S, Li L, et al (2018) Optimization of Molecules via Deep Reinforcement Learning. Nat Sci Reports 1–10. https://doi.org/10.1038/s41598-019-47148-x

14. Popova M, Isayev O, Tropsha A (2018) Deep reinforcement learning for de novo drug design. Sci Adv 4:. https://doi.org/10.1126/sciadv.aap7885

15. Gómez-Bombarelli R, Wei JN, Duvenaud D, et al (2018) Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. ACS Cent Sci 4:268–276. https://doi.org/10.1021/acscentsci.7b00572

16. Polykovskiy D, Artamonov A, Veselov M, et al (2019) Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. arXiv:181112823

17. Brown N, Fiscato M, Segler MHS, Vaucher AC (2019) GuacaMol: Benchmarking Models for de Novo Molecular Design. J Chem Inf Model 59:1096–1108. https://doi.org/10.1021/acs.jcim.8b00839

18. Bjerrum EJ (2017) SMILES Enumeration as Data Augmentation for Neural Network Modeling of Molecules. arXiv:170307076. https://doi.org/10.1016/j.energy.2015.05.143

19. Bjerrum EJ, Sattarov B (2018) Improving Chemical Autoencoder Latent Space and Molecular De Novo Generation Diversity with Heteroencoders. Biomolecules 8:1–13. https://doi.org/10.3390/biom8040131

20. Winter R, Montanari F, Noé F, Clevert DA (2019) Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. Chem Sci 10:1692–1701. https://doi.org/10.1039/c8sc04175j

21. Blaschke T, Olivecrona M, Engkvist O, et al (2018) Application of Generative Autoencoder in De Novo Molecular Design. Mol Inform 37:1–11.

https://doi.org/10.1002/minf.201700123

22. Winter R, Montanari F, Steffen A, et al (2019) Efficient Multi-Objective Molecular Optimization in a Continuous Latent Space Efficient Multi-Objective Molecular Optimization in a Continuous Latent Space. Chem Sci. https://doi.org/10.1039/c9sc01928f

23. Prykhodko O, Johansson S, Kotsias P-C, et al (2019) A De Novo Molecular Generation Method Using Latent Vector Based Generative Adversarial Network. ChemRxiv. https://doi.org/10.26434/chemrxiv.8299544

24. Lim J, Ryu S, Kim JW, Kim WY (2018) Molecular generative model based on conditional variational autoencoder for de novo molecular design. J Cheminform 10:1–9. https://doi.org/10.1186/s13321-018-0286-7

25. Wengong Jin,Regina Barzilay TSJ (2019) Multi-Resolution Autoregressive Graph-to-Graph Translation for Molecules. ChemRxiv 8266745:

26. Hochreiter S, Schmidhuber J (1997) Long Short-Term Memory. Neural Comput 9:1–32

27. Gaulton A, Hersey A, Nowotka ML, et al (2017) The ChEMBL database in 2017. Nucleic Acids Res 45:D945–D954. https://doi.org/10.1093/nar/gkw1074

28. Sun J, Jeliazkova N, Chupakin V, et al (2017) ExCAPE-DB: An integrated large scale dataset facilitating Big Data analysis in chemogenomics. J Cheminform 9:1–9. https://doi.org/10.1186/s13321-017-0203-5

29. (2019) MolVS: Molecule Validation and Standardization. In: https://molvs.readthedocs.io/en/latest/, v0.1.1

30. (2019) ExCAPEDB. In: https://solr.ideaconsult.net/search/excape/. Accessed 03/2019.

31. (2019) RDKit: Open-Source Cheminformatics Software. In: https://www.rdkit.org/

32. Butina D (1999) Unsupervised data base clustering based on daylight's fingerprint and Tanimoto similarity: A fast and automated way to cluster small and large data sets. J Chem Inf Comput Sci 39:747–750. https://doi.org/10.1021/ci9803381

33. Bjerrum EJ (2019) molvecgen: Molecular Vectorization and Batch Generation. In:

https://github.com/EBjerrum/molvecgen. Accessed 12/2018.

34. Pedregosa F, Varoquaux G, Gramfort A, et al (2012) Scikit-learn: Machine Learning in Python. 12:2825–2830

35. Chollet F (2019) Keras. In: https://keras.io/

36. Abadi M, Agarwal A, Barham P, et al (2016) TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. arXiv:160304467

37. Arora R, Basu A, Mianjy P, Mukherjee A (2016) Understanding Deep Neural Networks with Rectified Linear Units. arXiv:161101491

38. Williams RJ, Zipser D (1989) A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. Neural Comput 1:270–280. https://doi.org/10.1162/neco.1989.1.2.270

39. Kingma DP, Ba J (2014) Adam: A Method for Stochastic Optimization. arXiv:14126980 1–15

40. Wildman SA, Crippen GM (1999) Prediction of physicochemical parameters by atomic contributions. J Chem Inf Comput Sci 39:868–873. https://doi.org/10.1021/ci990307l

41. Bickerton GR, Paolini G V., Besnard J, et al (2012) Quantifying the chemical beauty of drugs. Nat Chem 4:90–98. https://doi.org/10.1038/nchem.1243

42. Tan C, Sun F, Kong T, et al (2018) A survey on deep transfer learning. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics) 11141 LNCS:270–279. https://doi.org/10.1007/978-3-030-01424-7_27

43. Preuer K, Renz P, Unterthiner T, et al (2018) Fréchet ChemNet Distance: A Metric for Generative Models for Molecules in Drug Discovery. J Chem Inf Model 58:1736–1741. https://doi.org/10.1021/acs.jcim.8b00234

44. Ester M, Kriegel H, Xu X, Miinchen D- (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise