Generative Recurrent Networks for *De Novo* Drug Design

Anvita Gupta,[a, b] Alex T. Müller,[a] Berend J. H. Huisman,[a] Jens A. Fuchs,[a] Petra Schneider,[a, c] and Gisbert Schneider*[a]

Abstract: Generative artificial intelligence models present a fresh approach to chemogenomics and *de novo* drug design, as they provide researchers with the ability to narrow down their search of the chemical space and focus on regions of interest. We present a method for molecular *de novo* design that utilizes generative recurrent neural networks (RNN) containing long short-term memory (LSTM) cells. This computational model captured the syntax of molecular representation in terms of SMILES strings with close to perfect accuracy. The learned pattern probabilities can be used for *de novo* SMILES generation. This molecular design concept eliminates the need for virtual compound library enumeration. By employing transfer learning, we fine-tuned the RNN's predictions for specific molecular targets. This approach enables virtual compound design without requiring secondary or external activity prediction, which could introduce error or unwanted bias. The results obtained advocate this generative RNN-LSTM system for high-impact use cases, such as low-data drug discovery, fragment based molecular design, and hit-to-lead optimization for diverse drug targets.

Keywords: Chemogenomics · deep learning · drug discovery · machine learning · medicinal chemistry

1 Introduction

Compound repositories of pharmaceutical companies contain up to a few million compounds. Even accounting for growth over time, these readily screenable libraries cover only a miniscule fraction of the synthetically accessible, druglike chemical space, which is estimated to contain $> 10^{30}$ molecules.[1] Because chemical space is too large to be screened in its entirety for drugs active against a particular target, automated design and screening of selected compounds with desired properties and likelihood of activity presents itself as a complementary approach. Computational *de novo* drug design involves exploring this vast chemical space for such compounds which may not have been synthesized before, and "deep learning" methods present concepts for chemical space navigation.[2] Here, we present a generative deep learning model based on recurrent neural networks (RNNs) for *de novo* drug design. We demonstrate the model's efficacy in three main use cases of *de novo* design: generating libraries for high-throughput screening, hit-to-lead optimization, and fragment-based hit discovery.

RNNs successfully solve machine learning tasks, such as natural language processing[3] and translation,[4] and composing music,[5] to name only a few domains. In particular, much of this success has been achieved by the use of recurrent networks of LSTM (*long short-term memory*) cells, first introduced by Hochreiter and Schmidhuber in 1997.[6] In the field of molecular informatics, RNNs based on LSTMs have been used to predict protein function from sequence[7] and successfully predict aqueous solubility of drug-like compounds.[8] RNNs were used as autoencoders to provide a latent representation of molecular structure for sampling preferred regions of chemical space.[9] Importantly, several research groups have recently demonstrated that RNNs can be employed to generate canonical SMILES strings, and can be fine-tuned by transfer learning.[10,11] In transfer learning, the machine learning model tries to keep information from a previously learned task to solve a different but related, yet unseen task.[12] Researchers at AstraZeneca have extended SMILES-generating RNNs by using this concept for reinforcement learning. The model's parameters were optimized to produce strings that scored highly according to an external scoring function. They applied this approach to generate sets of structures with low sulfur content, high predicted target activity, and other desirable properties.[13]

We here present a new approach to *de novo* drug design using RNN deep learning methodology (Figure 1). In

[a] A. Gupta, A. T. Müller, B. J. H. Huisman, J. A. Fuchs, P. Schneider, G. Schneider
Swiss Federal Institute of Technology (ETH), Department of Chemistry and Applied Biosciences, Vladimir–Prelog–Weg 4, 8093 Zurich, Switzerland
E-mail: gisbert@ethz.ch
[b] A. Gupta
Stanford University, Department of Computer Science, 450 Sierra Mall, Stanford, CA, 94305, USA
[c] P. Schneider
inSili.com GmbH, 8049 Zurich, Switzerland

the first part of this study, we train an LSTM-based RNN model to generate libraries of valid SMILES strings with high accuracy. We then use transfer learning to fine-tune our model, generating molecules that are structurally similar to drugs with known activities against particular targets, demonstrating for the first time that this approach is successful for "low-data" situations in early-phase drug design. Even with just a few representative molecules for model training, our approach yielded structures with similar chemical characteristics to known ligands.

In the second part of this study, we applied our generative model to fragment-based drug discovery by growing a library of leads starting from a known active fragment. To our knowledge, this represents the first time generative RNNs have been used for molecular design by fragment growing. Our deep learning model thus provides a fresh concept of generating general compound libraries, target-specific libraries (with both low and high amounts of training data), and bespoke focused libraries for fragment-based drug discovery.

2 Methods

2.1 Datasets

For training the RNN model, we compiled a dataset of 677,044 SMILES strings with annotated nanomolar activities ($K_{d/i/B}$, IC/EC$_{50}$) from ChEMBL22 (www.ebi.ac.uk/chembl). The dataset was then pre-processed to remove duplicates, salts and stereochemical information. In addition, pre-processing filtered out nucleic acids and long peptides which lay outside of the chemical space from which we sought to sample. The RNN was ultimately trained on 541,555 SMILES strings, with lengths from 34 to 74 SMILES characters (tokens).
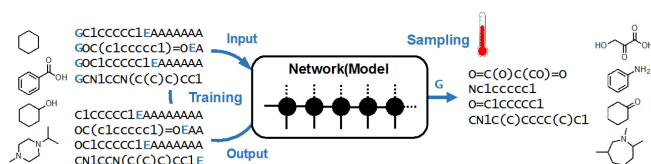
**Figure 1.** Schematic of model training (left) and compound design by sampling (right).

2.2 Model Structure

RNNs process a sequence of data $\mathbf{X} = x_1 x_2 x_n$ by taking as input each item x_i in the sequence. The RNN passes the input through a series of gates and returns some hidden state h_i and (optionally) an output vector $\hat{y}_i$. The hidden state h_i is passed from cell to cell, and reflects which
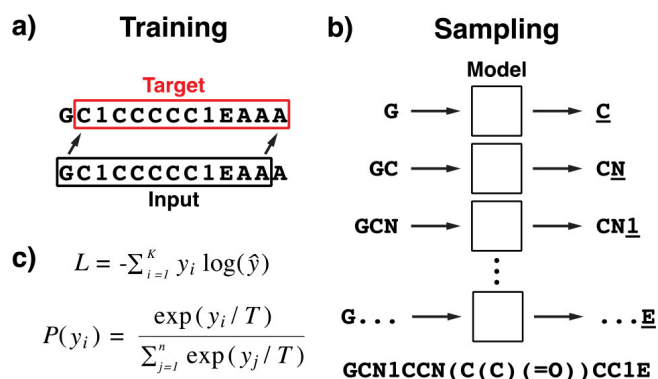
information the RNN has seen previously. Additional recurrent connections allow RNNs to learn complex temporal dependencies. In our model, the cells of the RNN belong to the class of LSTMs. LSTMs possess an input gate, a forget gate, and an output gate. Accordingly, LSTMs are able to specifically control what information passes to the next cell through the hidden state h_i. Important information can pass through successive cells unchanged. In this way, LSTMs solve the vanishing- or exploding-gradient-problem that RNNs experience due to backpropagation over long sequences.[14]

RNN models can be used to generate sequences one token at a time, as these models can output a probability distribution over all possible tokens at each time step. Typically, the RNN aims to predict the next token of a given input. It is worth noting that the input can be one or more tokens in length; if the input has m tokens then the model predicts the $(m+1)^{st}$ token. We trained the RNNs by maximum likelihood estimation. The target vector y_i is an array of one-hot encoded vectors, where each vector represents one token, and the output vector $\hat{y}_i$ is a probability distribution over the possible tokens In one-hot encoding, only one bit of a zero vector of the length of number of tokens in the dataset is set ("hot"). The model aims to maximize the probability assigned to the correct token for every vector in the array.

The structure of our model is shown in Figure 2. It consists of two LSTM layers, each with a hidden state vector of size 256, regularized with dropout. These two layers are followed by a dense output layer and a neuron unit with a *softmax* activation function. The input to the LSTM is a one-hot-encoded sequence of a molecule's SMILES string, where each string is split up into tokens. Each SMILES string is given a 'G' token (for "go") at the beginning, and an 'E' is
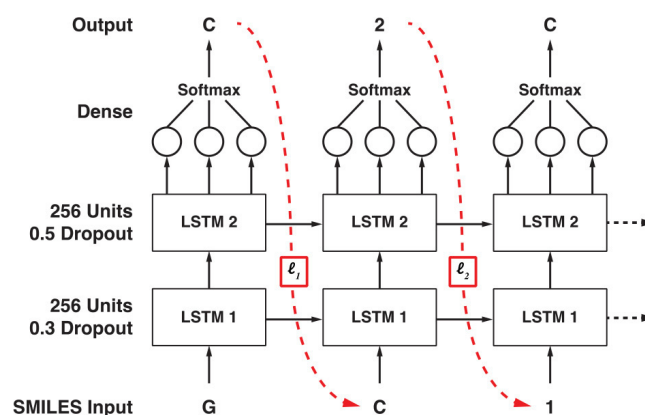
Figure 2. Model of the RNN–LSTM producing SMILES strings, token by token. The token 'G' denotes "GO" at the beginning of the SMILES string. During training, the model predicts the next token for each input token in the sequence. The loss L is calculated at each position as the categorical cross-entropy between the predicted and actual next token.

a) **Training**   **b)** **Sampling**

Target

G **C1CCCCC1EAAA**

GC1CCCCC1EAA A

Input

c)
$$L = -\sum_{i=1}^{K} y_i \log(\hat{y})$$

$$P(y_i) = \frac{\exp(y_i / T)}{\sum_{j=1}^{n} \exp(y_j / T)}$$

Model

G → □ → **C**

GC → □ → C**N**

GCN → □ → CN**1**

⋮

G . . . → □ → . . .**E**

GCN1CCN(C(C)(=O))CC1E

Figure 3. A) The training procedure for the final LSTM model. Each molecule was padded to the length *n* of the longest SMILES string (padding denoted by the token 'A'). The first *n*-1 characters were taken as the input, and the last *n*-1 characters were the target. B) Sampling procedure. The sentinel token 'G' was given to start. At every step of sampling, the last sampled character is taken as the next character in the generated sequence. Sampling continues until the token 'E' denoting "end of sequence" is generated. C) Equations for the calculation of the loss error L, and the *softmax* function $P(y_i)$ with temperature factor *T*.

added to denote the end of the SMILES string. The token 'A' was used for padding where needed.

2.3 Model Training and Sampling

We explored two methods for training the RNN. The first method was to break each input into overlapping windows of some length *l*, and predict the $l+1^{st}$ token of each window (Model 1). The loss was calculated from the likelihood of the $l+1^{st}$ token. The second method for training that we used, as shown in Figure 3a, pads every input string to *n* tokens, where *n* is the length of the longest SMILES string. For each token, the model predicts the next token in the sequence (Model 2). The loss was averaged over all the target tokens in all molecules.

When sampling from the model trained with method two, we fed the RNN only the sentinel token 'G' and sampled the next character from the predicted distribution. This next character was concatenated with the 'G'. Each time, we concatenated the last predicted letter to the hitherto generated sequence until the end token 'E' was produced (Figure 3b). When sampling characters from the model, we introduced an additional temperature parameter into the *softmax* function (Figure 3c). Higher sampling temperatures lead to greater structural diversity of the generated molecular structures but at the same time decrease the fraction of chemically valid SMILES strings, while lower temperatures lead to lower structural diversity but more conservative ("safer") predictions.[15]

2.4 Fine-tuning for Specific Ligand Subsets

After the model was trained to produce valid SMILES strings, we experimented with fine–tuning the model by further training on smaller subsets of selected compounds. The objective was to adapt the model to produce SMILES strings with higher similarity to these target-focused datasets. To simulate different early drug discovery scenarios, we tested the model's fine-tuning capabilities on three datasets of varying sizes: i) 4367 peroxisome proliferator-activated receptor gamma (PPARγ) inhibitors, ii) 1490 trypsin inhibitors, and iii) five structurally diverse transient receptor potential M8 (TRPM8) blockers. Ligands for each target were drawn from the ChEMBL data set. The sets of both trypsin and PPARγ inhibitors were pre-processed by removing stereochemistry and salts, and the central eighty percent of the molecules were selected by the length of their corresponding canonical SMILES strings. For TRPM8, the dataset of known inhibitors consisted of 448 compounds. These molecular structures were clustered by their Tanimoto similarity (MACCS keys; Molecular Operating Environment, The Chemical Computing Group, Montreal, Canada), yielding five clusters with distinct scaffolds. The most active compound was chosen as the representative molecule from each cluster. The network model was then fine–tuned on these five molecules.

After each training epoch, a sampled set of 100 molecules was generated. We measured the average Tanimoto similarity, comparing the sampled molecules and training data. The user was given the average and the percentage of duplicates in the sampled molecules. These two properties, distance and percentage duplicates, are often a trade-off. For trypsin and PPARγ, the model was fine-tuned for five epochs. For TRPM8, we fine-tuned for twelve epochs to compensate for the much smaller dataset. After tuning, we sampled 1000 SMILES for PPARγ and trypsin, and 100 SMILES for TRPM8, all at $T = 0.75$.

2.5 Fragment Growing Procedure

Instead of beginning sampling with the sentinel token 'G', we allowed the user to enter a fragment which they wish to be present in all SMILES generated. The RNN model will then read in and extend ("grow") the fragment SMILES, based on which tokens are likely to follow. It is worth noting that the fragment itself remains unmodified; we specifically tested the case where the fragment was at one end of the molecule, and provided exit vectors for the model to build upon.

2.6 Technical implementation

All deep learning models were implemented using Tensorflow (v1.2, www.tensorflow.org) and Keras (v2.0, https://keras.io) in Python (v3.6, www.python.org). SMILES string

validity and molecular feature calculation were carried out in RDkit (www.rdkit.org). The analysis of generated SMILES strings was performed using an iPython Notebook (v2.0, https://ipython.org). PCA was performed using the scikit-learn libraries (www.scikit-learn.org). We used MOE (v2016.08, Molecular Operating Environment, The Chemical Computing Group, Montreal) for clustering molecules for the low-data portion of this project.

3 Results and Discussion

### 3.1 Molecular Structure Generation

Based on the results of our preliminary experiment (Figure 4), we exclusively relied on RNN Model 2 for the productive runs. After training for twenty-two epochs, Model 2 (as described in section 2.2) produced an average of 98% valid SMILES strings at temperature $T = 0.5$ and an average of 70% valid SMILES strings at $T = 1.2$. Model 1 produced an average of 58% valid smiles strings at $T = 0.5$, and 30% valid structures at $T = 1.2$. Model 2 was thus selected for the productive runs.

Using Model 2, we sampled 30,107 SMILES, at $T = 0.75$. 93% of these SMILES were unique, and 92% of the unique

SMILES were valid SMILES. In order to compare the generated molecules to the original molecules used for RNN training, we calculated 24 common physiochemical features for the data. We performed a principal component analysis (PCA) on the features of the training data, and the newly generated molecules were transformed accordingly. Figure 5a shows the original and generated molecules plotted with respect to these principal components; we see that the generated molecules lie in the same space as the original molecules. Figure 5b specifically compares the distributions of molecular weight and clog$P$ for the generated and original molecules. We see that the medians and distributions are similar, although the generated molecules are skewed towards higher clog$P$ than the original molecules.

3.2 Target-specific Fine-tuning

After the model training on the 550,000 SMILES strings of bioactive compounds from ChEMBL, the trained Model 2

a)

b)
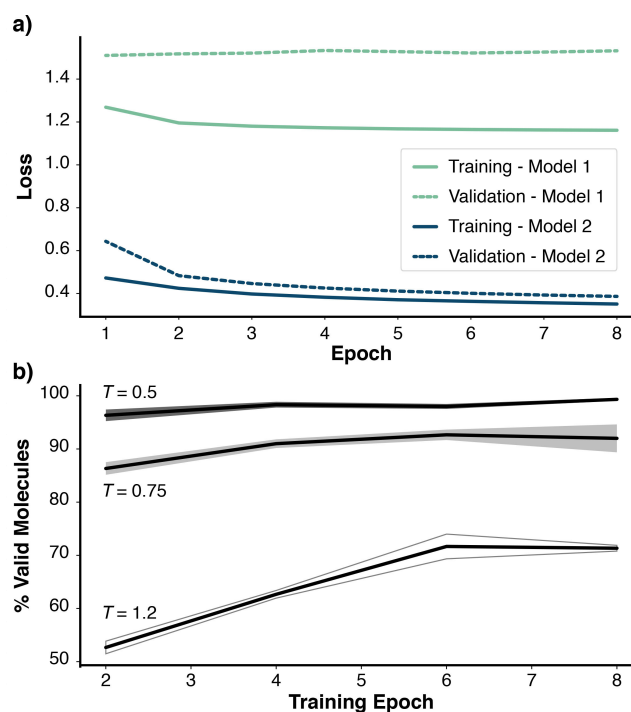
Figure 4. A) Training and Validation loss for Model 1 vs. Model 2. Model 2 was selected as the final model, and its sampling procedure is shown in Figure 2b. B) Percentage of valid molecules generated by Model 2 at different sampling temperatures (T). As the temperature increases, the percentage of valid molecules decreases.

a)

b)
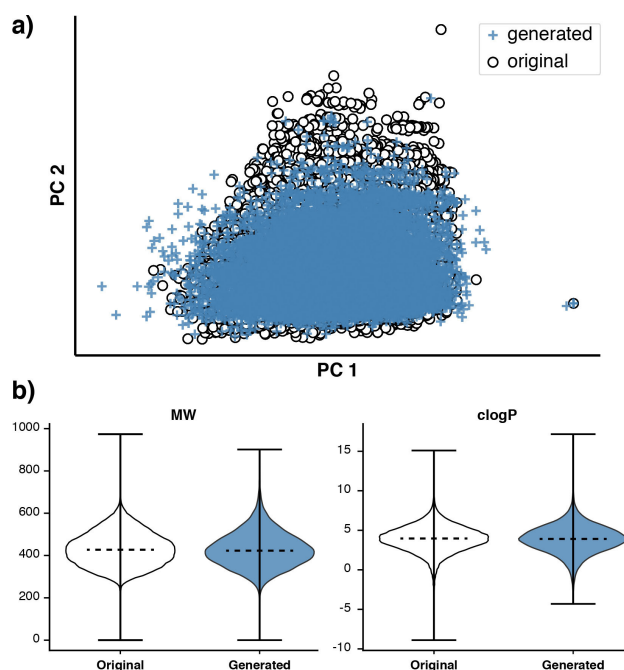
Figure 5. A set of 25,923 valid SMILES strings was generated from the trained Model 2, and 24 physiochemical features were calculated for the generated virtual molecules and the set of 550,000 original training molecules. A) PCA was performed on these 24 generated features from the training molecules, and the first two principal components (PC1, PC2) were selected. The coordinates of the generated molecules were transformed accordingly. We see overlap in the chemical subspace between these two sets of molecules. B) Violin–plots for molecular weight (MW) and clog$P$ distributions, with the medians shown as dashed lines. Visual inspection reveals a close match of the generated and original molecules.

was further fine-tuned for five epochs on 4367 PPARγ ligands. A set of 1000 molecules was generated after fine-tuning; 96% of these SMILES were valid. 90% of the valid SMILES strings were unique from each other, and 88% of the structures were unique from the known PPARγ ligands. Figure 6a shows that the generated PPARγ inhibitors lie in the same physicochemical subspace as the known PPARγ ligands. Here, the molecules are plotted with respect to the two highest components from PCA on the known ligands. The shift due to fine-tuning is visible in Figure 6b; as fine-tuning occurs, the generated molecules shift towards the part of the space that is most densely populated by known PPARγ ligands. As a quantitative measure of this shift, we calculated the average Tanimoto dissimilarity between the known PPARγ ligands and the generated molecules. Without fine tuning, the dissimilarity between the generated and known inhibitors was 0.425 ± 0.003 (*mean* $\pm$ *stderr.*). With fine-tuning, the dissimilarity between the generated and known molecules was 0.375 ± 0.003, a statistically significant decrease ($p < 0.0001$, one-sided t-test).

In order to test the model's ability to be optimized on datasets of diverse size, we then trained Model 2 (which had been given a warm-start on 550k SMILES strings) on 1490 known trypsin inhibitors. Again, we generated 1000 SMILES strings from the fine–tuned model; 93% of the SMILES were valid. 87% of the generated molecules were unique from each other, and 93% were not contained in the set of known trypsin inhibitors. Figure 6a shows that the fine-tuned molecules lie in the space of the known trypsin inhibitors, and Figure 6b illustrates that the generated molecules shifted in the space due to fine-tuning. All molecules are plotted with respect to the first two components from PCA of the known trypsin inhibitors. Without fine-tuning, the dissimilarity between the generated molecules and the known trypsin inhibitors was 0.440 ± 0.003 (*mean* $\pm$ *stderr.*). With fine-tuning, the dissimilarity decreased to 0.409 ± 0.003, again a statistically significant reduction ($p < 0.0001$, one-sided t-test).
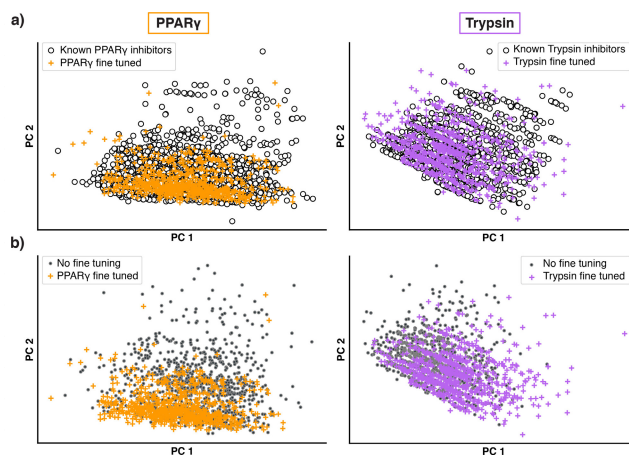
Modifications to known inhibitors are a key part of hit-to-lead optimization, as even small structural modifications can considerably affect the biological activity of known leads.[16] Figure 7 shows a selection of the five closest neighbors and five farthest neighbors from one known trypsin inhibitor. The closest molecules display small modifications along the rightmost aromatic ring. In addition, it is worth noting that these neighbors are displayed by Tanimoto similarity, which gives equal weight to all parts of the molecule. For practical applications, one may want to give higher weight to molecules that contain the same active fragment as the original lead, rather than generated molecules which are structurally similar in general.

These hypothetical use cases advocate our generative model as potentially useful for hit-to-lead optimization. We could demonstrate the ability of the approach to generate chemically valid structures within the models' respective applicability domain (as given by the properties of the training data).[17] Importantly, in contrast to other related RNN models,[10,11,13] ours does not rely on an explicit but limited SMILES vocabulary, which renders this new approach theoretically unlimited with regard to the chemical diversity of the training data. Model fine-tuning enabled the automated *de novo* design of target-focused compound sets, without the need of dedicated target prediction tools or other external scoring functions.

Figure 6. 1000 molecule structures were sampled after fine-tuning on sets of inhibitors of PPARγ (left) and trypsin (right). A) PCA was carried out on 24 physiochemical descriptors and fit to the set of original target inhibitors. The first two principal components (PC1, PC2) were selected for visualization. The molecules generated after RNN model fine-tuning are plotted together with the original ligands from ChEMBL. An analysis of the plots provides an idea of whether the fine-tuned molecules cover the space occupied by the original inhibitors. B) We plot the set of molecules generated without fine-tuning against the set of fine-tuned molecules. The axes are the same principal components as in panel a). We see a clear shift in the distributions of compounds generated with and without fine-tuning.

3.3 Fragment-growing

One main and novel use case of this generative RNN model is in fragment-based drug discovery (FBDD).[18] Instead of starting sampling with the sentinel token 'G', drug designers might want to start from a fragment known to bind to the target of interest. Our model can take the SMILES string of this fragment as input while sampling, and successively grow the remaining molecule.

For the minimalist thrombin–binding start fragment benzamidine (**12**)[19] shown in Figure 8, we illustrate how our generative model can be applied to FBDD. The fragment's exit vectors are shown as arrows. 1000 molecules were generated from the pre-trained RNN model which had been given a warm-start, of which 97% molecules were valid. All of these molecules contained the benzamidine fragment. Selected molecules are shown in Figure 8. As can

be seen, these generated molecules display structural diversity, and could be attractive for synthesis and testing. Consequently, our model can be used to generate compound libraries based on a single receptor-binding fragment. Furthermore, this approach could be fine-tuned
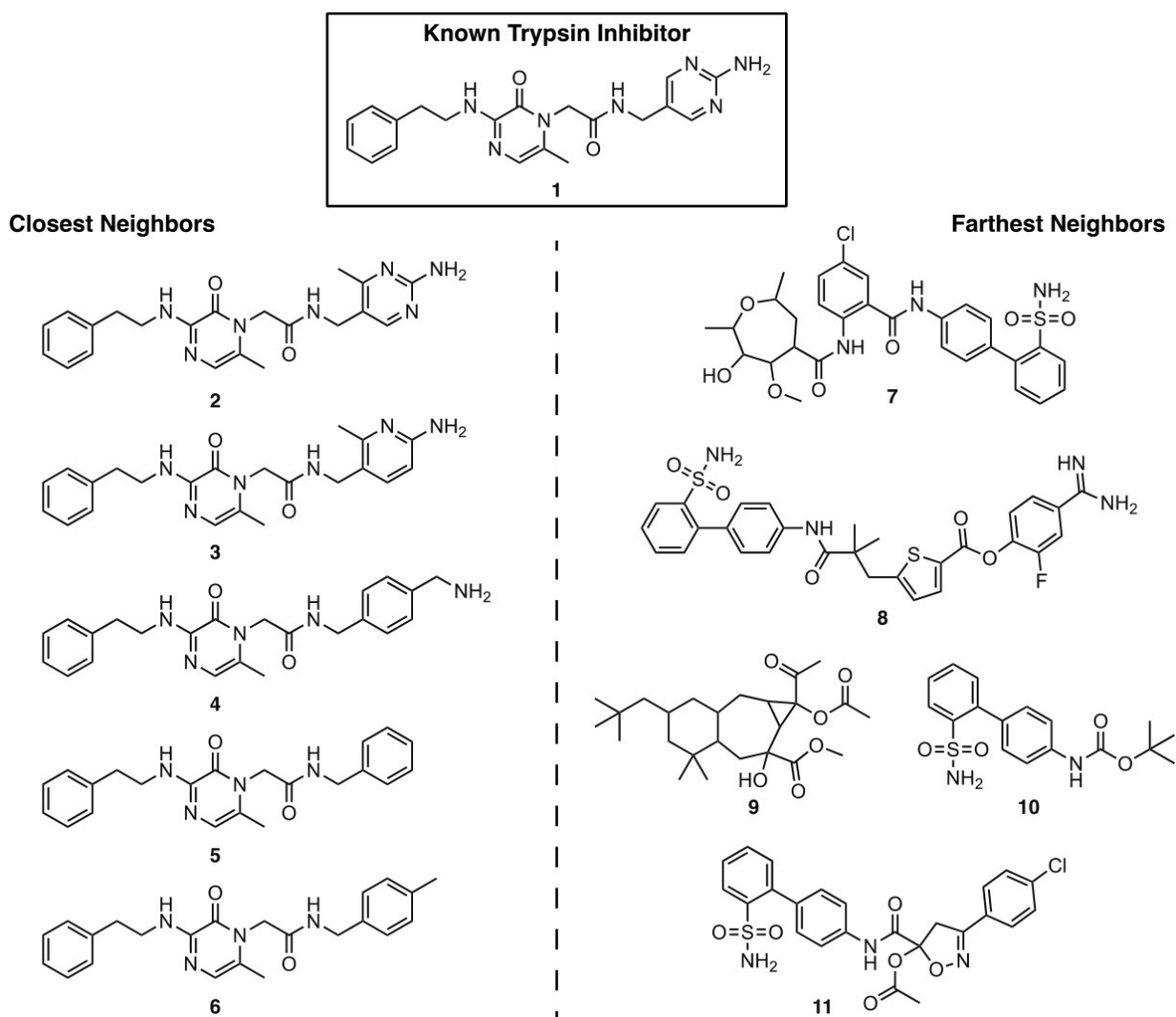
Figure 7. Left: Generated structures (2–6) with highest Tanimoto similarity to the known trypsin inhibitor (1). Right: Generated structures (7–11) with lowest Tanimoto similarity.
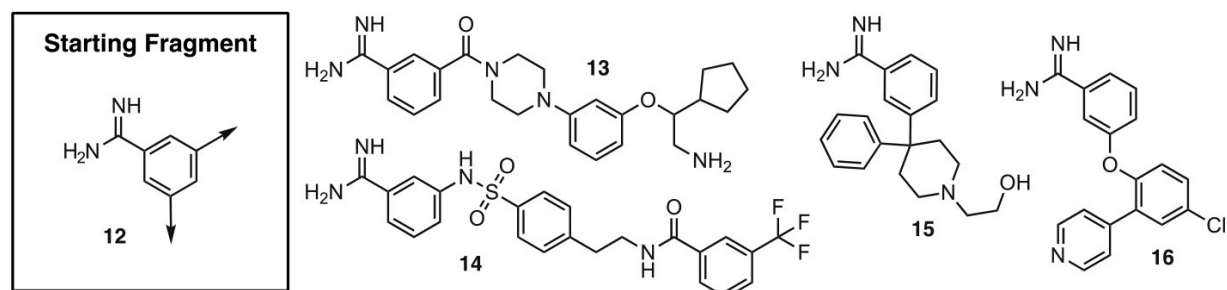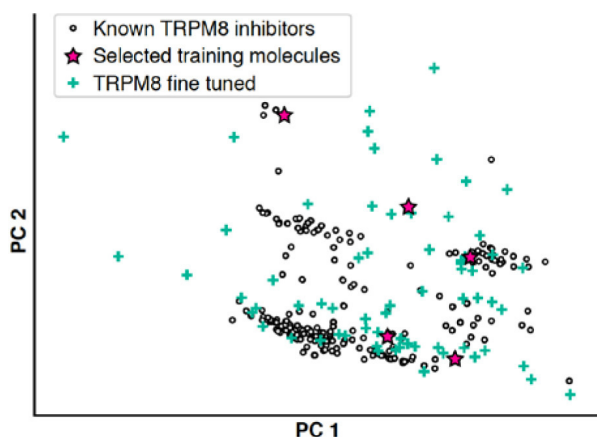

Figure 8. Molecules 13–16 were generated starting from two exit vectors (arrows) of the boxed fragment (benzamidine, 12). 12 has been shown to bind to the thrombin active site and may hence be well–suited for fragment–growing. 1000 molecules were sampled, and a collection of *de novo* generated molecules is shown on the right.

toward specific scaffolds or proprietary scaffold-centric compound libraries.

3.4 Low-data Drug Design

For several targets only a few ligands are known. This situation is characteristic for early-stage drug discovery projects, where *de novo* design may be especially useful. For this reason, we extended our generative model to the problem of molecular design in the presence of limited training data availability ("low-data"). Not only was the selected dataset small (five compounds), but the compounds were specifically chosen for the diversity of their scaffolds.
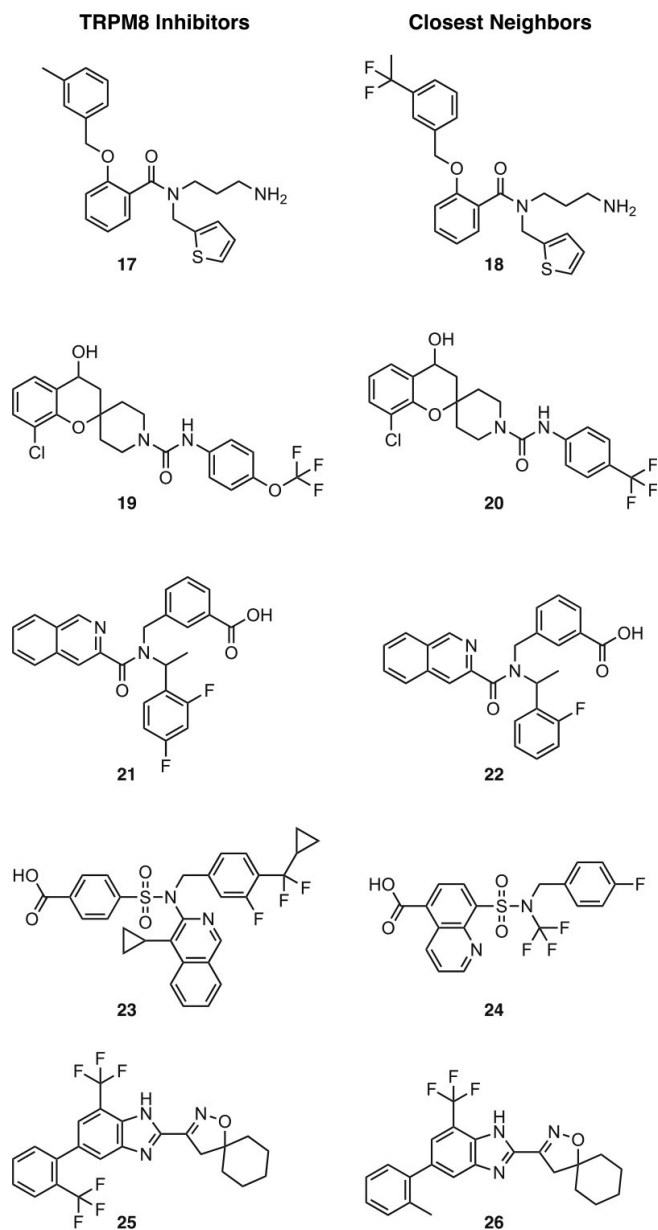
The five reference molecules were chosen by clustering the set of 448 known TRPM8 antagonists from ChEMBL22 by structural similarity, and choosing the most active ligand from each of the five clusters. Since TRPM8 had more actives than the ones taken in our full training set, we were able to examine how closely the generated structures approximated those of true TRPM8 inhibitors. Figure 9 shows the set of known TRPM8 antagonists, selected training compounds, and 100 *de novo* generated fine-tuned molecules in a PCA projection. All chemical structures are plotted by their scores of the first two principal components from PCA conducted on the known TRPM8 inhibitors. As can be seen, the generated molecules closely approximate four of the five reference compounds used for fine-tuning, and three of the existing TRPM8 clusters. However, one of the training compounds lies relatively far from the cluster of molecules it is supposed to represent. This is because the training compounds were chosen based on their activity, not based on which molecules lay in the center of their clusters.

We also see points that lie between two different clusters; indeed, several generated molecules combine structural motifs from different training ligands. Figure 10 shows each compound from the training set along with its closest generated neighbor. For instance, molecule **24** features the trifluoromethyl motif from compound **19** along with several structural motifs from compound **23**. Again,

Figure 9. A principal component analysis was conducted on 24 physicochemical features calculated on the full dataset of 448 known TRPM8 inhibitors. The five molecules that were chosen for fine-tuning the model are represented as stars in the coordinate system spanned by the first (PC 1) and second (PC 2) principal components. 100 molecules were sampled, and the positions of the valid molecules are indicated by green crosses.

Figure 10. The five TRPM8 inhibitors (17, 19, 21, 23, 25) used to fine-tune the RNN model are shown on the left. The respective generated molecule with highest Tanimoto similarity (nearest neighbor (18, 20, 22, 24, 26)), is shown on the right of every reference compound.

the closest neighbors were chosen by Tanimoto similarity (ECFP4 fingerprints). Out of the 100 generated fine-tuned molecules, 94 were not identical to one of five target molecules, even after 12 epochs of fine-tuning. 81 % of the generated molecules were unique.

4 Conclusions and Outlook

We have successfully applied a generative RNN-LSTM model for *de novo* design of chemical structures, and have demonstrated the model's applicability to (i) generating compound libraries for high-throughput screening, (ii) hit-to-lead optimization for targets, even with a small amount of data, and (iii) fragment-based drug discovery. The model successfully exhibited transfer learning; given a "warm-start", the model was able to be fine-tuned for specific subsets of compounds, with only a few epochs of additional training required. The structures generated with fine-tuning display significantly higher similarity to the respective reference molecules than the structures generated without fine-tuning. These molecule structures are largely unique and avoid overfitting the fine-tuning set of reference compounds.

We extend previous studies in several key ways, particularly in low-data drug design and fragment-based drug discovery. Transfer learning was shown to be successful for our approach, even when only a few known ligands were used for model fine-tuning. Although only a small number of representative molecules were chosen from the set of TRPM8 antagonists, the model generated molecules that lay within the chemical space of all TRPM8 antagonists, not only the ones used to train the model. In several cases, the generated molecules combine motifs from multiple known TRPM8 inhibitors. The model introduced several chemical modifications into these new compounds that could be useful for hit-to-lead optimization.

The application of the RNN model to fragment-growing could be useful in several situations. We demonstrated the usefulness of our model by on-the-fly generating a library of molecules containing a key fragment binding to thrombin. Importantly, this generative *de novo* design approach does not require extensive similarity searching or external scoring. The new molecular structures are generated instantly, which might be attractive for real-time *in situ* molecular modeling.

Our generative model itself contains fewer parameters than existing models, while achieving the same or improved percentage of valid molecules. Smaller models are often preferable to larger ones in deep learning because they have a reduced risk of overfitting. Indeed, our model shows little tendency of overfitting, even when trained for many epochs on low amounts of data.

This present approach does not strictly require an external scoring function for fine-tuning the parameters of the model. Instead, we optimize the parameters directly from chemical structures possessing some desirable property, thereby avoiding the risk of potentially error-prone scoring. A downside of this method is the necessity for available active ligands for parameter optimization. A further limitation of our current approach is the necessity for model fine-tuning over a particular number of epochs, in order to avoid generating compound duplicates. We attempted to circumvent this issue by allowing the user to make the decision on when to stop training; after every epoch of fine-tuning, the software tool provides the user with the percentage of duplicates in the molecules generated, and how similar the generated molecules are to the provided subset. These two quantities are often a trade-off, and we currently request the user to make the final decision, rather than applying an arbitrary rule to all fine-tuning sets.

Our approach might be particularly useful when combined with some *a priori* knowledge of a specific active fragment that should be kept constant. However, this current method cannot grow molecules in more than one direction (exit vector) from the start fragment. The RNN-LSTM model was originally trained on the SMILES of bioactive, synthesizable molecules from ChEMBL, and the introduced solutions appear to be intrinsically informed by synthesizability. We sought to avoid the need for a scoring function based on synthesizability that would introduce additional error to the model. We are currently evaluating the prospective practical applicability of the new design approach in hit- and lead-finding projects, contributing to exploring the opportunities and limitations of automated drug discovery.[20]

Conflict of Interest

P. S. and G. S. declare a potential financial conflict of interest in their role as life–science industry consultants and cofounders of inSili.com GmbH, Zurich.

References

[1] a) C. Lipinski, A. Hopkins, *Nature* **2004**, *432*, 855–861; b) G. Schneider, U. Fechner, *Nat. Rev. Drug Discov.* **2005**, *4*, 649–663; c) H. J. Verheij, *Mol. Divers.* **2006**, *10*, 377–388; d) J. L. Reymond, *Acc. Chem. Res.* **2015**, *48*, 722–730.

[2] a) Y. LeCun, Y. Bengio, G. Hinton, *Nature* **2015**, *521*, 436–444; b) S. Ekins, *Pharm. Res.* **2016**, *33*, 2594–2603.; c) E. Gawehn, J. A. Hiss, G. Schneider, *Mol. Inf.* **2016**, *35*, 3–14; d) I. I. Baskin, D. Winkler, I. V. Tetko, *Expert Opin. Drug Discov.* **2016**, *11*, 785–795.

[3] A. Veliz-Cuba, H. Z. Shouval, K. Josić, Z. P. Kilpatrick, *J. Comput. Neurosci.* **2015**, *39*, 235–254.

[4] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Ł. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, J. Dean, *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*, arXiv:1609.08144, **2016**.

[5] Y. C. Subakan, P. Smaragdis, Diagonal RNNs in Symbolic Music Modeling, in: *EEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA) 2017*, arXiv:1704.05420, **2017**.

[6] S. Hochreiter, J. Schmidhuber, *Neural Comput.* **1997**, *9*, 1735–1780.

[7] a) J. Lanchantin, R. Singh, B. Wang, Y. Qi, *Pac. Symp. Biocomput.* **2016**, *22*, 254–265; b) X. L. Liu, C. A. Xueliang, L. Liu, *Deep Recurrent Neural Network for Protein Function Prediction from Sequence*, bioRxiv **2017**, doi.org/10.1101/103994.

[8] A. Lusci, G. Pollastri, P. Baldi, *J. Chem. Inf. Model.* **2013**, *53*, 1563–1575.

[9] R. Gómez–Bombarelli, D. Duvenaud, J. Miguel Hernández-Lobato, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, A. Aspuru-Guzik, *Automatic Chemical Design Using a Data-driven Continuous Representation of Molecules*, arXiv:1610.02415, **2016**.

[10] M. H. S. Segler, T. Kogej, C. Tyrchan, M. P. Waller, *Generating Focussed Molecule Libraries for Drug Discovery with Recurrent Neural Networks*, arXiv:1701.01329, **2017**.

[11] E. J. Bjerrum, R. Threlfall, *Molecular Generation with Recurrent Neural Networks (RNN*, arXiv:1705.04612, **2017**.

[12] M. E. Taylor, P. Stone, *J. Machine Learning Res.* **2009**, *10*, 1633–1685.

[13] M. Olivecrona, T. Blaschke, O. Engkvist, H. Chen, *J. Cheminform.* **2017**, *9*, 48.

[14] J. Wang, L. Zhang, Q. Guo, Z. Yi, *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, doi: 10.1109/TNNLS.2017.2677968.

[15] P. Reverdy, N. E. Leonard, *IEEE Trans. Autom. Sci. Eng.* **2016**, *13*, 54–67.

[16] J. P. Hughes, S. Rees, S. B. Kalindjian, K. L. Philpott, *Br. J. Pharmacol.* **2011**, *162*, 1239–1249.

[17] M. Mathea, W. Klingspohn, K. Baumann, *Mol. Inf.* **2016**, *35*, 160–180.

[18] D. A. Erlanson, S. W. Fesik, R. E. Hubbard, W. Jahnke, H. Jhoti, *Nat. Rev. Drug Discov.* **2016**, *15*, 605–619.

[19] J. Stürzebecher, H. Vieweg, P. Wikström, D. Turk, W. Bode, *Biol. Chem. Hoppe Seyler* **1992**, *373*, 491–496.

[20] P. Schneider, G. Schneider, *J. Med. Chem.* **2016**, *59*, 4077–4086.