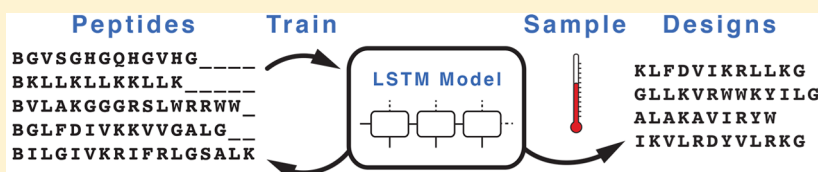# Recurrent Neural Network Model for Constructive Peptide Design

Alex T. Müller, Jan A. Hiss, and Gisbert Schneider*

Swiss Federal Institute of Technology (ETH), Department of Chemistry and Applied Biosciences, Vladimir-Prelog-Weg 4, CH−8093 Zurich, Switzerland

S *Supporting Information*

ABSTRACT: We present a generative long short-term memory (LSTM) recurrent neural network (RNN) for combinatorial de novo peptide design. RNN models capture patterns in sequential data and generate new data instances from the learned context. Amino acid sequences represent a suitable input for these machine-learning models. Generative models trained on peptide sequences could therefore facilitate the design of bespoke peptide libraries. We trained RNNs with LSTM units on pattern recognition of helical antimicrobial peptides and used the resulting model for de novo sequence generation. Of these sequences, 82% were predicted to be active antimicrobial peptides compared to 65% of randomly sampled sequences with the same amino acid distribution as the training set. The generated sequences also lie closer to the training data than manually designed amphipathic helices. The results of this study showcase the ability of LSTM RNNs to construct new amino acid sequences within the applicability domain of the model and motivate their prospective application to peptide and protein design without the need for the exhaustive enumeration of sequence libraries.

INTRODUCTION

Recent advances in applied machine learning have supported human creativity for molecular design. In this context, the umbrella term "constructive machine learning" describes an entire class of problem-solving methods for which the ultimate learning goal is not necessarily to find the optimal model of the training data but rather to identify new instances (e.g., molecules) from within the applicability domain of the model that are likely to exhibit the desired properties.[1−3] In contrast to models that are typically used to classify a given set of unlabeled domain instances post hoc, constructive machine learning models are generative and able to capture an infinite or exponentially large combinatorial search space. This property is relevant and potentially promising for molecular de novo design and inverse QSAR modeling.[4,5] Although the overall concept of constructive modeling is not new, there has been considerable recent interest in this and related "deep learning" approaches, which is partly motivated by the availability of customized computer hard- and software solutions.[6−8]

Long short-term memory (LSTM) models belong to the class of recurrent neural networks (RNNs) incorporating so-called memory units. LSTM systems partly overcome the problem of vanishing or exploding gradients in backpropagation training of conventional RNNs.[9] Recurrently connected LSTM memory cells share a constant weight matrix, span several time steps, and therefore allow for constant error flow through the network.[10] Unlike hidden Markov models[11] or *n*-gram approaches,[12] RNNs are generally able to capture long-range dependencies in natural language, handwriting, speech, or music.[10,13−15] With the incorporated LSTM memory cells, RNNs can even learn long-range sequence correlations.[16,17]

Proteins and peptides can be represented in terms of sequential amino acid residues, so we hypothesized that learning their grammar may be possible with recurrent LSTM networks for the purpose of generative de novo sequence design. RNNs have already experienced a rediscovery as models for natural language processing.[18] Given a sequence of words, these systems predict a distribution of the possible next words. For example, if such a model receives the input sequence "Let's meet for a", it will assign probabilities for possible next words. In this case, "coffee", "beer", or "barbecue" will likely be assigned high probabilities in contrast to words such as "car", "book", or "glasses". The same concept may be applied to amino acid patterns in protein and peptide sequences. Provided that a sufficiently diverse and representative training pool is available, explicit feature selection of meaningful molecular descriptors would not be strictly required anymore, as the model directly learns the most appropriate internal representation for a given problem directly from the amino acid sequence.[19]

Recurrent LSTM systems have already been successfully applied to protein sequence analysis. Examples include the application of recurrent neural networks with or without LSTM memory units for protein secondary structure prediction,[20−22] protein homology detection,[23] and protein localization in subcellular compartments.[24] The learning systems in these
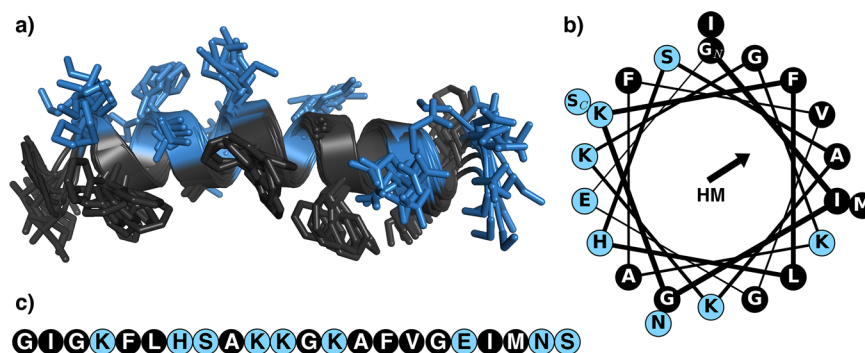
Figure 1. Structural organization of amino acid side chains in the antimicrobial peptide magainin 2. (a) NMR structural ensemble of magainin 2 in the presence of dipalmitoylphosphatidylcholine (DPC) micelles (PDB ID: 2mag).[26] The hydrophobic amino acid side chains (black) face toward the hydrophobic core of the micelles (not shown) while aligning the polar residues (blue) toward the aqueous medium. (b) Idealized helical wheel plot of magainin 2 (generated with the modlAMP Python package, version 3.3.0[27]). The degree of spatial separation of the hydrophobic from polar amino acids can be quantified by the hydrophobic moment (HM). The associated HM vector points toward the hydrophobic face of the helix. (c) Sequence representation of magainin 2 in single letter code (black circle: hydrophobic residue; blue circle: polar residue). This peptide features alternating clusters of hydrophobic and polar amino acids.
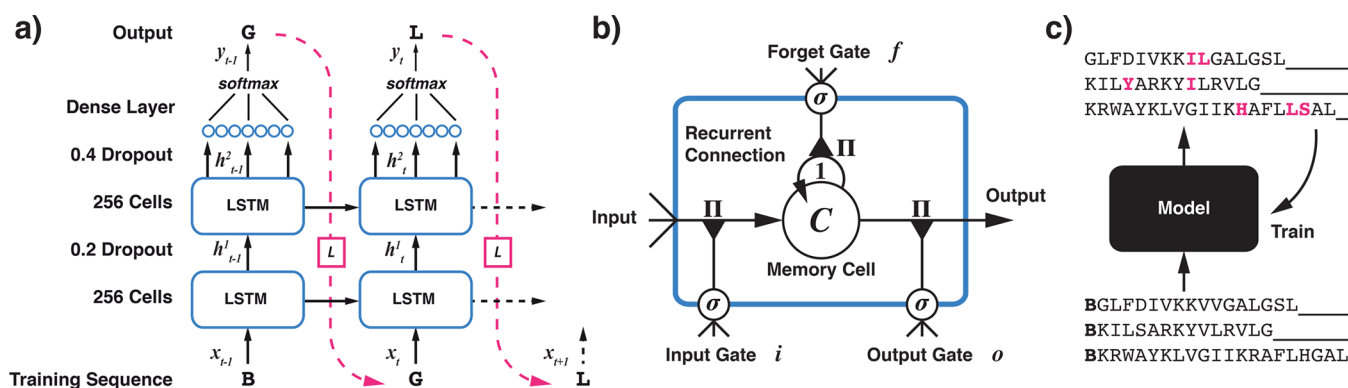

Figure 2. Schematic of the LSTM RNN architecture (adapted from ref 34). Information flow is from bottom to top and left to right. (a) Unfolded representation of the network. In each training step t, the network receives a one-hot vector-encoded amino acid from a training sequence as input. The computed output y is compared to the actual amino acid to calculate the categorical cross-entropy loss (red dashed line, L, eq 1). Through backpropagation, the weight matrix (shared by all time steps) is adapted. Note that the second LSTM layer forwards all states to a densely connected network layer, which transforms the information back to a 22-size one-hot vector y through the *softmax* function. The token 'B' invokes the trained network model for sequence generation (i.e., for actual peptide design). Every subsequently generated output amino acid is rejoined with the preceding one, thereby obtaining a growing residue sequence. (b) Structure of one LSTM unit with a forget gate, as described by Gers et al.[35] The input, forget, and output gates control the flow of information into and out of the memory cell. All gates are fed with data input as well as output of the preceding units. The constant weight, denoted by **1**, allows for information propagation over multiple time steps. $\prod$ denotes the function of the gates to multiply the information flow with their state value. If a gate's state value is zero, no information will flow. (c) Simplified depiction of the network training process. All input sequences are padded with space characters to the length of the longest peptide in the training set, and a "begin" token (B) is added at the start. For every residue, the following amino acid is predicted by the model and joined. This generated sequence is then compared to the actual amino acids to calculate training loss and adapt the network weights.

studies mainly vary in the applied model architecture, but all systems are decision machines (classifiers) and have not been used for de novo sequence generation.

In the present study, we introduce LSTM RNNs for amino acid sequence design and analyze their generative potential. For proof-of-concept, we selected antimicrobial peptides (AMPs) for model training. Certain AMPs directly affect the integrity of the bacterial cell membrane, leading to membrane pore formation and membranolysis.[25] Here, we focus on linear cationic peptides forming amphipathic helices. Some of these sequences feature repetitive residue motifs, enabling the formation of organized peptide complexes in bacterial membranes (Figure 1). Amphipathicity was shown to be a relevant property for their antimicrobial activity.[25] We expected RNNs to capture this fundamental feature. Once trained, we

invoked the system to generate new, potentially amphipathic AMPs.

■ METHODS

Training Data. We collected peptide sequences from the following three publicly accessible sources: "a database for antimicrobial peptides" (ADAM),[28] "antimicrobial peptide database" (APD),[29] and "database of Anuran defense peptides" (DADP).[30] The ADAM database contains sets of sequences for different secondary structures from which we extracted the helical AMPs (cluster ID: AC_003). From the APD, we retrieved all sequences annotated as "helical". We included the entire DADP peptide database. The majority of all of these AMPs form linear helical structures unless they contain Cys residues potentially forming disulfide bridges.[31−33] Sequences containing Cys were removed accordingly, as were sequences

incorporating unnatural amino acids, both for ease of potential later synthesis. The final training set consisted of 1554 peptides encompassing 7−48 amino acid residues with an average sequence length of 20.8 ± 7.7 (mean ± SD) and median = 21 residues.

Model Structure and Training. We used the recurrent network structure shown in Figure 2 for model training and peptide design. Figure 2b illustrates the incorporated adapted LSTM unit with an additional forget gate, as proposed by Gers et al.[35] The unit input combines the information from the input layer of the current input x_t at time step t and from the hidden layer h_{t-1} signals of the previous step $t − 1$ (Figure 2a). The summed weighted input is further processed with a *tanh* activation function. The sigmoidal input gate receives the same information as the input node (Figure 2b). It controls the amount of information that is passed from the input node to the memory cell. This cell, also called the internal state,[9] has a recurrent connection with a constant weight of 1 ("constant error carousel").[10] The forget gate[35] acting on this connection allows the network to flush the content of its internal state. Finally, we obtain the output of each node as the product of the cell memory and the sigmoidal output gate.

For sequence generation, the amino acid sequence is unknown a priori, rendering the use of bidirectional RNNs unfeasible.[36] Therefore, we trained a two-layered unidirectional LSTM RNN with 256 memory units per layer. The output of the second LSTM layer was fed into a densely connected feed-forward layer with 22 output neurons, combining the output signals with a *softmax* function. A constant bias of 1 was added to the LSTM forget gate, as recommended by Jozefowicz et al.,[37] and the LSTM layers were regularized by 20 and 40% dropout for layers one and two, respectively, to reduce the risk of overfitting.[38,39] We used the Adam optimizer[40] with a learning rate of 0.01 on the categorical cross-entropy loss function (eq 1). The categorical cross-entropy loss L between the computed and the actual target vectors of the network was calculated for every one-hot encoded residue in a sequence of K amino acid symbols (single letter code) as

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

where $\hat{y}_i$ denotes the computed ith one-hot residue vector and y_i the actual ith target amino acid vector in the training data. To obtain suitable hyperparameters for model development, we performed five-fold cross-validation with different model architectures over 200 epochs each. The number of LSTM cells was chosen from [24, 32, 48, 64, 128, 256, 512] for one or two layers. We used dropout to regularize all layers with a fraction of 0.1- or 0.2-times the layer number. For every architecture, we determined the training epoch at which minimal validation loss was observed. The validation loss at this state of each network served as the criterion for selecting the best performing architecture overall. In every fold of cross-validation, the model weights were reinitialized with a different random seed to prevent residual knowledge from prior folds. We implemented the network model in Python using the Keras library[41] (version 2.0.2) with the TensorFlow[42] (version 1.3.0) backend. Peptide sequence handling was managed with the modlAMP[27] package (version 3.3.0). Network training and sequence generation were performed on an NVIDIA GeForce GTX 1080 Ti graphics processing unit. For an overview of the workflow, see Figure 3.
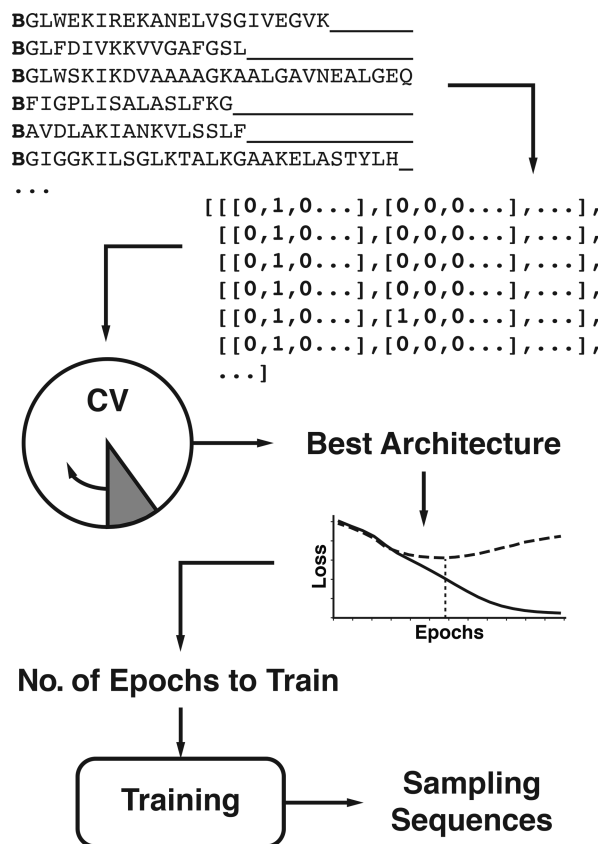

Figure 3. Overview of the workflow applied in this study. Amino acid sequences were first padded with space characters to the length of the longest sequence, and the N-terminal token 'B' was added to denote the "begin" of a peptide. One-hot encoding then transformed the sequences to a machine-readable format. A grid search of different model architectures through five-fold cross-validation (CV) was performed. The network architecture with the lowest validation loss was chosen as the final model. For this architecture, the optimal number of training epochs (before over-fitting was observed) was determined, and the final model was trained on the complete data set for the so-obtained number of epochs. Novel sequences were sampled from this final model.

Data Processing. To allow for generating arbitrary new sequences, we added the token 'B' (for "begin") to the N-terminus of all amino acid sequences. To simplify data handling and bring all sequences to the same length, we additionally padded the sequences with tailing space characters to the length of the longest sequence in the training set (here, 48 residues). All sequences were represented by a one-hot encoding scheme based on binary vectors with a length equal to the size of the vocabulary.[10] Our applied vocabulary of possible letters (feature vector size) consisted of 20 natural amino acid letters plus 'B' ("begin" token) and "space" (padding token). The final sequence data matrix contained 1554, 48, and 22 elements, respectively (number of sequences, padded length, and feature vector length). The targets for network training (i.e., solving the loss function) were defined as the respective next amino acid for each position in the input, e.g., for the sequence GLFDIVK_, the corresponding target is LFDIVK__.

Sequence Generation. Sampling of new sequences from the final trained network was performed for 2000 cycles. Each cycle was invoked through the start character 'B'. Sampling was continued until either a padding space character or the maximal

sequence length (48 residues) was reached. To control sequence variability while sampling, we introduced a temperature factor into the *softmax* function (eq 2).[34] The temperature-controlled probability P of picking amino acid y at sequence position i is defined as

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

(2)

Evaluation of Generated Sequences. We analyzed the de novo-generated sequences according to the following criteria:

(i) Comparison to the training data. We determined the percentage of valid sequences (without 'B') and of unique novel peptides compared to the training sequences. Global peptide descriptor values were then compared between the valid sequences and the training set and analyzed for statistically significant differences. In addition, we calculated the Euclidean distances from the training set in the combined global peptide descriptor space.

(ii) Predicted antimicrobial activity. We evaluated the generated sequences using the CAMP AMP prediction tool (URL http://camp.bicnirrh.res.in/predict).[43] The predicted *pseudo*-probabilities of valid sequences being AMPs were compared to the predictions of the training set.

In both approaches (i) and (ii), the generated peptides were additionally compared to randomly shuffled sequences ("Random") with the same amino acid distribution as the training set as well as to manually generated, presumably amphipathic, helical sequences ("Helices"). Generating amphipathicity was achieved by placing basic residues every three to four amino acids and filling the gaps with hydrophobic amino acids (implemented as `sequences.Helices` in the modlAMP Python package).[27]

■ RESULTS

We implemented and trained LSTM RNNs on modeling peptide sequences with antimicrobial activity. We then used the best-performing model for generative peptide de novo design. The general workflow of sequence padding and one-hot encoding model cross-validation and final training is shown in Figure 3. Five-fold cross-validation on different network architectures revealed that neither small nor large networks performed best. The validation error did not fully converge for simple RNN models with only few layers and neurons (Supporting Information). Increasing the network size to more than two layers with 256 neurons led to rapid over-fitting of the training data distribution. On the basis of these preliminary experiments, we selected a network architecture with two layers containing 256 neurons each for the productive runs. The cross-validation and training performance of this network, measured as categorical cross-entropy loss (eq 1), is summarized in Figure 4. An average validation loss of 0.56 ± 0.06 (mean ± SD) was obtained after 167 epochs. We chose this number of epochs for training the network on the complete data set. The evolution of training loss for the final network is shown in Figure 4b. The final state of the model was saved and used for sampling novel sequences.

In preliminary peptide design runs, we tested the effect of different sampling temperatures (*softmax* function, eq 2) on the validity of the constructed sequences (Figure 5a). High temperatures led to more diverse sequences, whereas low
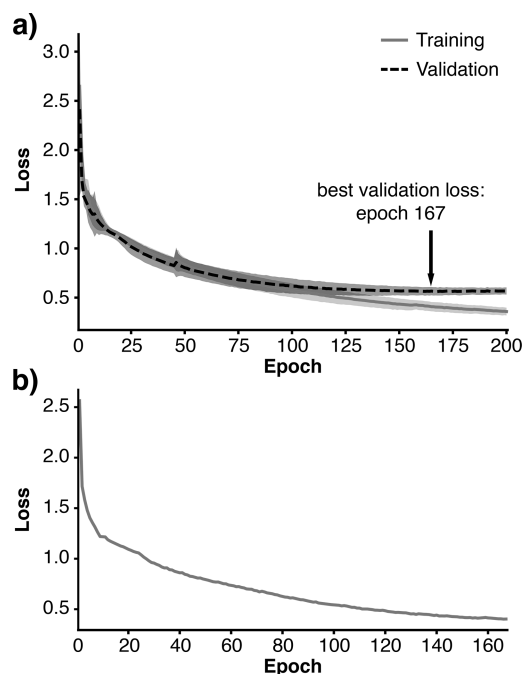

Figure 4. Loss evolution in five-fold cross-validation (a) and final training (b). The loss was computed as the categorical cross-entropy for both the training (gray solid) and validation (black dashed) sets (eq 1). (a) Five-fold cross-validation results. The average validation error has a minimum around 167 epochs. Shaded areas represent standard deviations. (b) The network was retrained for 167 epochs on all available sequence data.

temperatures resulted in peptides with limited sequence diversity. At a temperature of 1.25, the sequence validity started to converge, and we decided to productively sample at this temperature value.

Constructive generation of 2000 sequences by the network model yielded 1747 unique peptides, which were nonidentical to the training data ("valid"). Of the newly generated sequences, 88% had a length between seven and 48 residues (19 ± 8 (mean ± SD)) corresponding to the extremes of the training data distribution.

To allow for a direct comparison of the designed sequences with the training set, we randomly selected the same number of newly generated sequences as in the training set (1554) for further investigation. Our analysis of the main features of the generated sequences enabled a more detailed comparison to the training data set (Figure 5). To check whether the system did not just default to generating sequences with a certain amino acid distribution, we additionally created a set of 1554 *pseudo*-random peptide sequences (7−48 residues) with the same amino acid distribution as the training set. As an additional set, we generated cationic amphipathic helices with high hydrophobic moments (0.65 ± 0.10). All four sequence sets were disparate. A statistical analysis of the differences in major peptide features between the training data and the generated sequences is presented in Table 1. A pairwise Welch's *t*-test was used to test the null hypothesis that the average observed min-max scaled feature values are identical between the training and generated sequences. The *t*-test indicated statistically significant differences for all examined peptide features (*p*-value < 0.05) except for the sequence length, charge density, and isoelectric point.
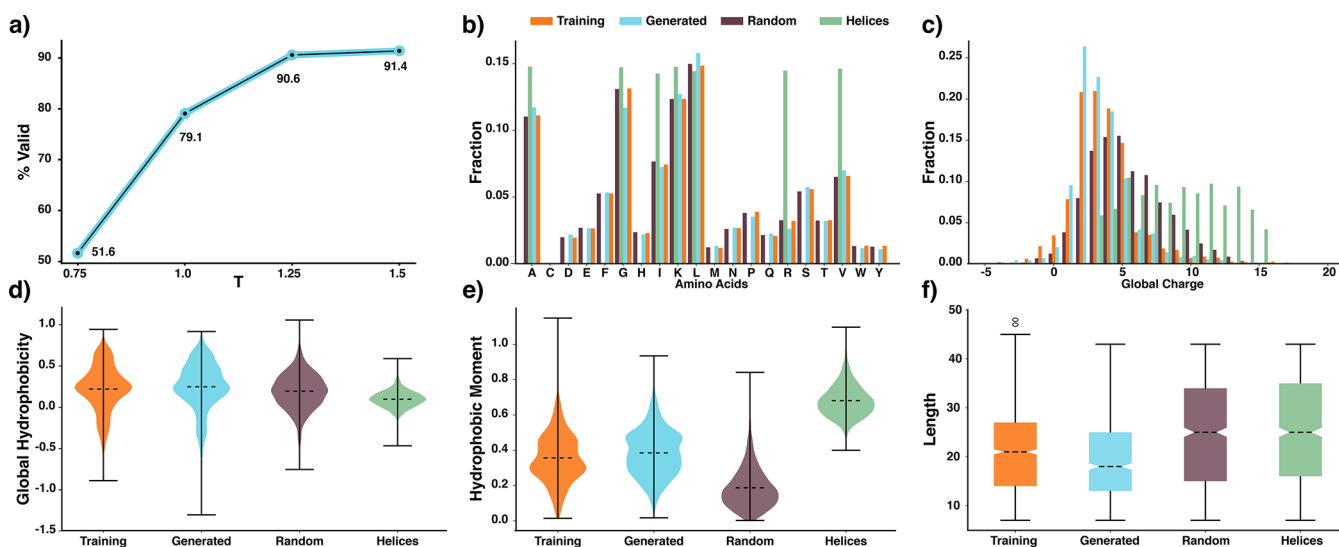
Figure 5. Comparison of the main peptide features between the training data (Training, orange), the generated sequences (Generated, blue), the *pseudo*-random sequences with the same amino acid distribution as the training set (Random, purple), and the manually created presumed amphipathic helices (Helices, green). The horizontal dashed lines represent the mean (violin plots) and median (box plots) values, and the whiskers extend to the most extreme nonoutlier data points. (a) Percentages of valid unique sequences obtained from sampling 1000 sequences at different temperatures, (b) amino acid frequencies, (c) total charge, (d) Eisenberg hydrophobicity, (e) Eisenberg hydrophobic moment, and (f) sequence length. The figure was generated with modlAMP's `GlobalAnalysis.plot_summary` method in Python.[27]

Table 1. Results of Welch's *t*-Test between the Min-Max Scaled Descriptor Distributions of the Training Set and the Generated Sequences[a]

feature	training	generated	*p*-value
charge	3.2 ± 2.3	3.0 ± 1.9	0.004
length	20.8 ± 7.7	20.4 ± 7.7	0.101
molecular weight	2226 ± 809	2159 ± 805	0.022
charge density	0.0014 ± 0.0009	0.0014 ± 0.0008	0.341
Eisenberg hydrophobicity	0.22 ± 0.29	0.26 ± 0.27	0.0004
Eisenberg hydrophobic moment	0.36 ± 0.14	0.38 ± 0.13	0.00003
isoelectric point	11.4 ± 2.0	11.3 ± 1.7	0.666
aromaticity	0.091 ± 0.078	0.085 ± 0.075	0.045

[a]Values are given as the mean ± SD of the unscaled descriptors for the training and generated sequences. Features were calculated with the modlAMP Python package (documentation: https://modlamp.org).[27] The statistical analysis was performed with the `scipy.stats` Python package.[57]

Finally, we used all descriptors from Table 1 to calculate the Euclidean distance from the sampled sequences to the training set in this combined descriptor space. The sampled sequences show a Euclidean distance of 0.5 ± 0.3 (mean ± SD). As a comparison, the same distance calculation was performed for the random (0.7 ± 0.3) and helical (0.8 ± 0.3) sequence sets. The calculated distances revealed that our model generated sequences with a significantly greater similarity to the training data compared to those of the two comparison sets (*p*-value < 0.05, Welch's *t*-test).

The model-sampled sequences visually approximated the amino acid distribution of both the training and random sets with a slightly increased fraction of hydrophobic amino acids (Figure 5b). This increase is also reflected in the global hydrophobicity distribution when comparing these three sets (Figure 5d, Table 1). However, the spatial orientations of hydrophobic and polar amino acids differ, leading to

significantly increased hydrophobic moments (*p*-value < 0.001, two-sided Welch's *t*-test) compared to those of the random set (Figure 5e). With an average value of 0.38 ± 0.13 (mean ± SD), the hydrophobic moment of the sampled sequences was also higher than that of the training set (0.36 ± 0.14). A histogram depicting the distribution of hydrophobic moments in both sets is shown in Figure S3.

To quantify the antimicrobial potential of the generated peptides, we predicted their activity with a publicly accessible AMP prediction tool from the CAMP[44] server. We chose the CAMP random forest classifier because it performed superiorly to other tools in a recent benchmark study.[45] The *pseudo*-probabilities of the antimicrobial (*P*(AMP)) peptides were obtained from this random forest classifier model (Figure 6). Of all generated sequences, 82% were predicted to be "active" (*P*(AMP) ≥ 0.5), and 18% of the generated sequences were
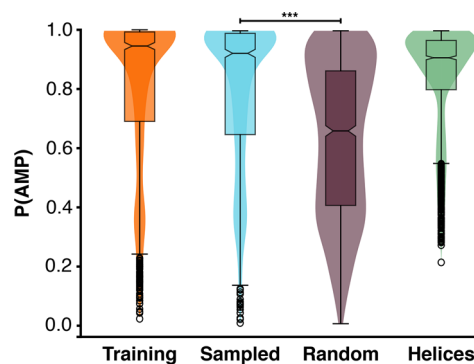

Figure 6. *Pseudo*-probabilities of sequences predicted to be AMPs (CAMP random forest classifier[44]). Training (orange): training data (known AMPs); Sampled (blue): sequences newly generated by the LSTM RNN model; Random (purple): *pseudo*-random sequences with the same amino acid probabilities as the training data; Helices (green): presumed amphipathic helices with repetitive basic residues and hydrophobic patches. ***p*-value < 0.001 (two-sided Welch's *t*-test).
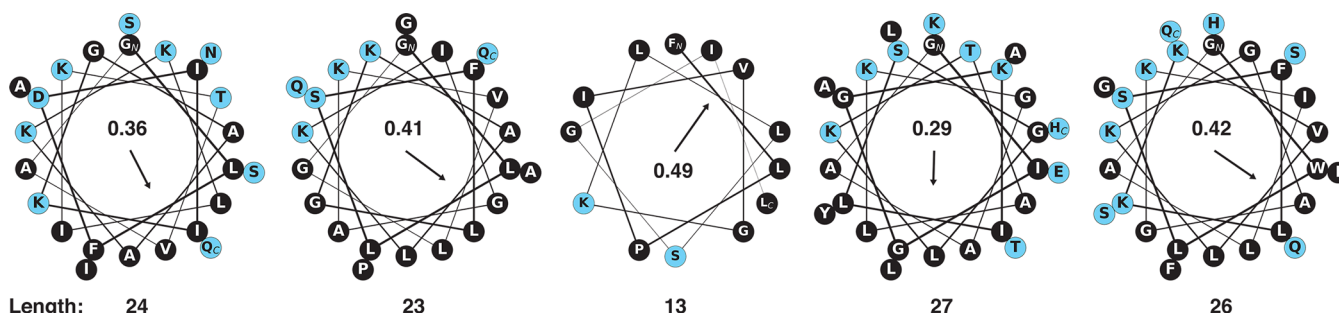
Figure 7. Helical wheel plots of five top-ranking peptides with $P(AMP) = 1$ from the sequences generated by the LSTM RNN model. Predictions of the CAMP random forest classifier[43] were sorted to select the top-ranking sequences. The numbers and corresponding arrows depict Eisenberg's hydrophobic moment[46] for the idealized α-helical conformation. The corresponding sequence length is given below the wheels.

predicted to be "inactive" ($P(AMP) < 0.5$). Welch's t-test rejected the hypothesis that the random and sampled data sets have equal means of activity predictions (p-value < 0.01). According to the AMP predictions, our LSTM RNN model generated new peptides with a significantly higher probability of being active AMPs than the random sequences possessing the training set's amino acid frequencies (Figure 6). To further challenge the AMP prediction tool, we analyzed presumably helical amphipathic sequences. The corresponding $P(AMP)$ predictions exceeded the values obtained for the training data with an average *pseudo*-probability of being AMP of 0.83 ± 0.18 compared to 0.80 ± 0.27 (training), 0.79 ± 0.25 (sampled), and 0.63 ± 0.26 (random).

To obtain a visual interpretation of the top predicted sequences from the sampled set, we generated helical wheel plots for the peptides considered top ranking by the CAMP prediction model (Figure 7). The plots suggest the formation of amphipathic helical structures similar to the magainin 2 example depicted in Figure 1.

■ DISCUSSION

The present study demonstrates the first application of an LSTM RNN to de novo sequence design using AMPs as an example. The approach conceptually differs from template-based peptide design strategies[47] and random sequence generation methods, where amino acids are drawn from a predefined distribution and subsequently concatenated.[48] Because the LSTM RNN model relies on its internal high-dimensional sequence representation, it should not merely reproduce specific sequence templates but potentially constructs "fuzzified" versions of the sequences present in the training data.[49] This concept is related to the "simulated molecular evolution" method, which is based on a stochastic sequence evolution process.[50]

The results obtained suggest that the applied LSTM RNN model was able to generate an internal representation of helical AMPs. Although we invoked every generation process by the same initial 'B' token, the network model did not merely reproduce the training sequences but revealed its own internal interpretation of the data. This is reflected in the fact that none of the generated sequences is identical to a training peptide. Although certain general peptide properties differ from the training data distribution, the sequence generation with RNNs performed significantly better in approximating the AMP sequence space compared to random or rule-based peptide design in the global peptide descriptor space. The global charge of the generated sequences was observed to be mainly positive, corroborating earlier findings that AMPs obtain their bacterial

membrane targeting ability through a subtle balance of charge interactions and hydrophobicity.[51]

The main difference between the network-generated sequences and the random sequences is their amphipathicity, which is exemplified by a higher hydrophobic moment for the network-generated set (Figure 5e). The hydrophobic moment translates back to a regular pattern of charged and hydrophobic residues at specific locations in the sequence, which must have been learned by the network. We hypothesize that this also gave rise to the better predictions of the CAMP models. To shed light on this assumption, we manually created presumed amphipathic helical sequences with alternating positively ionizable and hydrophobic amino acids. These sequences received even higher scores by the CAMP prediction than the training data (Figure 6). This result corroborates our hypothesis that the LSTM RNN actually learned to recognize the grammar of amphipathicity in peptides.

It has been shown that a high hydrophobic moment alone is an insufficient criterion for potent and selective antimicrobial activity.[52,53] It is reasonable to assume that a certain degree of conformational flexibility is needed to obtain targeted membranolytic activity.[54−56] Consequently, generating "fuzzified" versions of the training sequences will prove useful for peptide design and optimization. Importantly, the sequence variations introduced by the network model are not random but follow the learned training data variability. Evidently, LSTM RNNs are restricted to the applicability domain covered by the training data.[58,59] From a purely sequence-oriented vantage point, extrapolation is difficult, as peptide sequences represent discrete points in chemical space without smooth intermediate transitions. This fact highlights the advantage of a continuous descriptor space that allows for continuous transitions between molecules and, to a limited extent, enables extrapolation.[60−62] Combining LSTM RNNs with predictive models trained on molecular descriptors could therefore be suitable for the design of bespoke activity-focused amino acid sequence libraries.

■ CONCLUSIONS AND OUTLOOK

The results of this study advocate the use of RNNs with LSTM for generative machine learning of amino acid sequence data and peptide de novo design. The network models were shown to generate peptide libraries of a desired size within the applicability domain of the model. This positive result motivates the application of LSTM RNNs to diverse peptide design tasks. From the outcome of this study, we conclude that the approach may be best suited for use in combination with predictive models evaluating the quality of the generated sequences. In future studies, this concept may be extended to

address recent advances in convolutional networks for image generation, so-called generative adversarial networks (GANs).[54] In GANs, a generator network produces data for subsequent scrutiny by a second discriminator network to check if the newly generated data originate from the training domain. The generator constantly adapts toward "tricking" the discriminator to believe that the generated data are actual training data. As such, the generated data are gradually morphed to mimic the training domain of interest. This idea and the question of whether some of the peptides generated in this present study are biologically functional are the subjects of current research.

■ ASSOCIATED CONTENT

Ⓢ Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/acs.jcim.7b00414.

> Table S1, Cross-validation results for all tested architectures; Figures S1 and S2, Loss curves of all cross-validation runs; Figure S3, Histograms of hydrophobic moments; a Python code template for model training and sequence generation is available from https://github.com/alexarnimueller/LSTM_peptides (PDF)
>
> S2, Sequences used for model training; S3, Sequences generated by the model; S4, Random sequence library; S5, Helical sequence library (ZIP)

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: gisbert@ethz.ch.

ORCID Ⓘ

Gisbert Schneider: 0000-0001-6706-1084

Author Contributions

All authors jointly designed the research, evaluated the results, and wrote the manuscript. A.T.M. performed the experiments. All authors have given approval to the final version of the manuscript.

Notes

Several references point to non-peer-reviewed texts and preprints. These partly inspired this present work and are cited to account for the actuality of the topic of this article.

The authors declare the following competing financial interest(s): G.S. is a co-founder of inSili.com LLC, Zurich, and a consultant in the life science industry.

■ ABBREVIATIONS

LSTM, long short-term memory; RNN, recurrent neural network; AMP, antimicrobial peptide

■ REFERENCES

(1) Colton, S.; de Mantaras, R. L.; Stock, O. Computational Creativity: Coming of Age. *AI Mag.* **2009**, *30* (3), 11−14.

(2) Liu, L.; Tang, L.; Dong, W.; Yao, S.; Zhou, W. An Overview of Topic Modeling and Its Current Applications in Bioinformatics. *SpringerPlus* **2016**, *5* (1), 1608.

(3) White, D.; Wilson, R. C. Generative Models for Chemical Structures. *J. Chem. Inf. Model.* **2010**, *50* (7), 1257−1274.

(4) Schneider, P.; Schneider, G. De Novo Design at the Edge of Chaos. *J. Med. Chem.* **2016**, *59* (9), 4077−4086.

(5) Miyao, T.; Funatsu, K. Finding Chemical Structures Corresponding to a Set of Coordinates in Chemical Descriptor Space. *Mol. Inf.* **2017**, *36* (8), e1700030.

(6) Miotto, R.; Wang, F.; Wang, S.; Jiang, X.; Dudley, J. T. Deep Learning for Healthcare: Review, Opportunities and Challenges. *Briefings Bioinf.* **2017**, bbx044.

(7) Gawehn, E.; Hiss, J. A.; Schneider, G. Deep Learning in Drug Discovery. *Mol. Inf.* **2016**, *35* (1), 3−14.

(8) LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521* (7553), 436−444.

(9) Hochreiter, S.; Schmidhuber, J. Long Short−Term Memory. *Neural Comput.* **1997**, *9* (8), 1735−1780.

(10) Lipton, Z. C.; Berkowitz, J.; Elkan, C. A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv:1506.00019*, 2015.

(11) Baum, L. E.; Petrie, T. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *Ann. Math. Stat.* **1966**, *37* (6), 1554−1563.

(12) Brown, P. F.; DeSouza, P. V.; Mercer, R. L.; Della Pietra, V. J.; Lai, J. C. Class-Based N-Gram Models of Natural Language. *Comput. Linquistics* **1992**, *18* (4), 467−479.

(13) Graves, A.; Mohamed, A.−R.; Hinton, G. Speech Recognition with Deep Recurrent Neural Networks. *2013 IEEE Int. Conf. Acoust. Speech Signal Process* **2013**, No. 6, 6645−6649.

(14) Graves, A.; Liwicki, M.; Fernandez, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. A Novel Connectionist System for Improved Unconstrained Handwriting Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *31* (5), 855−868.

(15) Boulanger-Lewandowski, N.; Bengio, Y.; Vincent, P. Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription. *Proc. 29th Int. Conf. Mach. Learn.* **2012**, 1159−1166.

(16) Sutskever, I.; Martens, J.; Hinton, G. Generating Text with Recurrent Neural Networks. *Proc. 28th Int. Conf. Mach. Learn.* **2011**, 1017−1024.

(17) Segler, M. H. S.; Kogej, T.; Tyrchan, C.; Waller, M. P. Generating Focussed Molecule Libraries for Drug Discovery with Recurrent Neural Networks. *ACS Cent. Sci.* **2017**, DOI: 10.1021/acscentsci.7b00512.

(18) Goldberg, Y. A Primer on Neural Network Models for Natural Language Processing. *J. Artif. Intell. Res.* **2016**, *57*, 345−420.

(19) Qian, N.; Sejnowski, T. J. Predicting the Secondary Structure of Globular Proteins Using Neural Network Models. *J. Mol. Biol.* **1988**, *202* (4), 865−884.

(20) Heffernan, R.; Yang, Y.; Paliwal, K.; Zhou, Y. Capturing Non-Local Interactions by Long Short-Term Memory Bidirectional Recurrent Neural Networks for Improving Prediction of Protein Secondary Structure, Backbone Angles, Contact Numbers and Solvent Accessibility. *Bioinformatics* **2017**, *33* (18), 2842−2849.

(21) Baldi, P.; Pollastri, G. The Principled Design of Large-Scale Recursive Neural Network Architectures-DAG-RNNs and the Protein Structure Prediction Problem. *J. Mach. Learn. Res.* **2003**, *4*, 575−602.

(22) Sønderby, S. K.; Winther, O. Protein Secondary Structure Prediction with Long Short Term Memory Networks. *arXiv:1412.7828*, 2014.

(23) Hochreiter, S.; Heusel, M.; Obermayer, K. Fast Model-Based Protein Homology Detection without Alignment. *Bioinformatics* **2007**, *23* (14), 1728−1736.

(24) Sønderby, S. K.; Sønderby, C. K.; Nielsen, H.; Winther, O. Convolutional LSTM Networks for Subcellular Localization of Proteins. In *Algorithms for Computational Biology. AlCoB 2015. Lecture Notes in Computer Science*; Springer, 2015; Vol. 9199, pp 68−80.

(25) Fjell, C. D.; Hiss, J. A.; Hancock, R. E. W.; Schneider, G. Designing Antimicrobial Peptides: Form Follows Function. *Nat. Rev. Drug Discovery* **2012**, *11* (1), 37−51.

(26) Gesell, J.; Zasloff, M.; Opella, S. J. Two-Dimensional 1H NMR Experiments Show That the 23-Residue Magainin Antibiotic Peptide Is an α-Helix in Dodecylphosphocholine Micelles, Sodium Dodecylsulfate Micelles, and Trifluoroethanol/water Solution. *J. Biomol. NMR* **1997**, *9* (2), 127−135.

(27) Müller, A. T.; Gabernet, G.; Hiss, J. A.; Schneider, G. modlAMP: Python for Antimicrobial Peptides. *Bioinformatics* **2017**, *33* (17), 2753−2755.

(28) Lee, H.−T.; Lee, C.−C.; Yang, J.−R.; Lai, J. Z. C.; Chang, K. Y. A Large-Scale Structural Classification of Antimicrobial Peptides. *BioMed Res. Int.* **2015**, *2015*, e475062.

(29) Wang, G.; Li, X.; Wang, Z. APD3: The Antimicrobial Peptide Database as a Tool for Research and Education. *Nucleic Acids Res.* **2016**, *44* (D1), D1087−D1093.

(30) Novković, M.; Simunić, J.; Bojović, V.; Tossi, A.; Juretić, D. DADP: The Database of Anuran Defense Peptides. *Bioinformatics* **2012**, *28* (10), 1406−1407.

(31) Rozek, T.; Wegener, K. L.; Bowie, J. H.; Olver, I. N.; Carver, J. A.; Wallace, J. C.; Tyler, M. J. The Antibiotic and Anticancer Active Aurein Peptides from the Australian Bell Frogs Litoria Aurea and Litoria Raniformis. *Eur. J. Biochem.* **2000**, *267* (17), 5330−5341.

(32) Wong, H.; Bowie, J. H.; Carver, J. a. The Solution Structure and Activity of Caerin 1.1, an Antimicrobial Peptide from the Australian Green Tree Frog, Litoria Splendida. *Eur. J. Biochem.* **1997**, *247* (2), 545−557.

(33) Haney, E. F.; Hunter, H. N.; Matsuzaki, K.; Vogel, H. J. Solution NMR Studies of Amphibian Antimicrobial Peptides: Linking Structure to Function? *Biochim. Biophys. Acta, Biomembr.* **2009**, *1788* (8), 1639−1655.

(34) Gupta, A.; Müller, A. T.; Huisman, B. J. H.; Fuchs, J. A.; Schneider, P.; Schneider, G. Generative Recurrent Networks for De Novo Drug Design. *Mol. Inf.* **2017**, *36*, e1700111.

(35) Gers, F. A.; Schmidhuber, J.; Cummins, F. Learning to Forget: Continual Prediction with LSTM. *Neural Comput.* **2000**, *12* (10), 2451−2471.

(36) Schuster, M.; Paliwal, K. K. Bidirectional Recurrent Neural Networks. *IEEE Trans. Signal Process.* **1997**, *45* (11), 2673−2681.

(37) Jozefowicz, R.; Zaremba, W.; Sutskever, I. An Empirical Exploration of Recurrent Network Architectures. *Proc. 32nd Int. Conf. Mach. Learn. Lille, Fr.* **2015**, *37*.

(38) Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929−1958.

(39) Gal, Y.; Ghahramani, Z. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. *Adv. Neural Inf. Process. Syst.* **2016**, 1019−1027.

(40) Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*, 2014.

(41) Chollet, F. *Keras.* GitHub 2015. https://github.com/fchollet/keras/.

(42) Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G. S.; Davis, A.; Dean, J.; Devin, M. et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv:1603.04467*, 2016.

(43) Waghu, F. H.; Gopi, L.; Barai, R. S.; Ramteke, P.; Nizami, B.; Idicula-Thomas, S. CAMP: Collection of Sequences and Structures of Antimicrobial Peptides. *Nucleic Acids Res.* **2014**, *42* (D1), 1154−1158.

(44) Waghu, F. H.; Barai, R. S.; Gurung, P.; Idicula-Thomas, S. CAMP R3: A Database on Sequences, Structures and Signatures of Antimicrobial Peptides: Table 1. *Nucleic Acids Res.* **2016**, *44* (D1), D1094−D1097.

(45) Gabere, M. N.; Noble, W. S. Empirical Comparison of Web-Based Antimicrobial Peptide Prediction Tools. *Bioinformatics* **2017**, *33* (13), 1921−1929.

(46) Eisenberg, D.; Weiss, R. M.; Terwilliger, T. C. The Helical Hydrophobic Moment: A Measure of the Amphiphilicity of a Helix. *Nature* **1982**, *299* (5881), 371−374.

(47) Wang, G.; Hanke, M. L.; Mishra, B.; Lushnikova, T.; Heim, C. E.; Chittezham Thomas, V.; Bayles, K. W.; Kielian, T. Transformation of Human Cathelicidin LL−37 into Selective, Stable, and Potent Antimicrobial Compounds. *ACS Chem. Biol.* **2014**, *9* (9), 1997−2002.

(48) Müller, A. T.; Kaymaz, A. C.; Gabernet, G.; Posselt, G.; Wessler, S.; Hiss, J. A.; Schneider, G. Sparse Neural Network Models of Antimicrobial Peptide-Activity Relationships. *Mol. Inf.* **2016**, *35* (11−12), 606−614.

(49) Graves, A. Generating Sequences With Recurrent Neural Networks. *arXiv:1308.0850*, 2013.

(50) Schneider, G.; Wrede, P. The Rational Design of Amino Acid Sequences by Artificial Neural Networks and Simulated Molecular Evolution: De Novo Design of an Idealized Leader Peptidase Cleavage Site. *Biophys. J.* **1994**, *66* (2), 335−344.

(51) Wieprecht, T.; Dathe, M.; Beyermann, M.; Krause, E.; Maloy, W. L.; MacDonald, D. L.; Bienert, M. Peptide Hydrophobicity Controls the Activity and Selectivity of Magainin 2 Amide in Interaction with Membranes. *Biochemistry* **1997**, *36* (20), 6124−6132.

(52) Stutz, K.; Müller, A. T.; Hiss, J. A.; Schneider, P.; Blatter, M.; Pfeiffer, B.; Posselt, G.; Kanfer, G.; Kornmann, B.; Wrede, P.; Altmann, K.−H.; Wessler, S.; Schneider, G. Peptide−Membrane Interaction Between Targeting and Lysis. *ACS Chem. Biol.* **2017**, *12* (9), 2254−2259.

(53) Lin, Y.−C.; Lim, Y. F.; Russo, E.; Schneider, P.; Bolliger, L.; Edenharter, A.; Altmann, K.−H.; Halin, C.; Hiss, J. a.; Schneider, G. Multidimensional Design of Anticancer Peptides. *Angew. Chem., Int. Ed.* **2015**, *54* (35), 10370−10374.

(54) Epand, R. M.; Epand, R. F. Lipid Domains in Bacterial Membranes and the Action of Antimicrobial Agents. *Biochim. Biophys. Acta, Biomembr.* **2009**, *1788* (1), 289−294.

(55) Matsuzaki, K. Control of Cell Selectivity of Antimicrobial Peptides. *Biochim. Biophys. Acta, Biomembr.* **2009**, *1788* (8), 1687−1692.

(56) Huang, Y.; He, L.; Li, G.; Zhai, N.; Jiang, H.; Chen, Y. Role of Helicity of a-Helical Antimicrobial Peptides to Improve Specificity. *Protein Cell* **2014**, *5* (8), 631−642.

(57) McKinney, W. Data Structures for Statistical Computing in Python. *Proc. 9th Python Sci. Conf.* **2010**, 1697900 (Scipy), 51−56.

(58) Mathea, M.; Klingspohn, W.; Baumann, K. Chemoinformatic Classification Methods and Their Applicability Domain. *Mol. Inf.* **2016**, *35* (5), 160−180.

(59) Baskin, I. I.; Winkler, D.; Tetko, I. V. A Renaissance of Neural Networks in Drug Discovery. *Expert Opin. Drug Discovery* **2016**, *11* (8), 785−795.

(60) Schneider, G.; Schuchhardt, J.; Wrede, P. Artificial Neural Networks and Simulated Molecular Evolution Are Potential Tools for Sequence-Oriented Protein Design. *Bioinformatics* **1994**, *10* (6), 635−645.

(61) Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Networks. *Adv. Neural Inf. Process. Syst.* **2014**, *2*, 2672−2680.

(62) Hiss, J. A.; Stutz, K.; Posselt, G.; Weßler, S.; Schneider, G. Attractors in Sequence Space: Peptide Morphing by Directed Simulated Evolution. *Mol. Inf.* **2015**, *34* (11−12), 709−714.