nature machine intelligence

ARTICLES

https://doi.org/10.1038/s42256-019-0017-4

Feedback GAN for DNA optimizes protein functions

Anvita Gupta[1] and James Zou[1,2*]

Generative adversarial networks (GANs) represent an attractive and novel approach to generate realistic data, such as genes, proteins or drugs, in synthetic biology. Here, we apply GANs to generate synthetic DNA sequences encoding for proteins of variable length. We propose a novel feedback-loop architecture, feedback GAN (FBGAN), to optimize the synthetic gene sequences for desired properties using an external function analyser. The proposed architecture also has the advantage that the analyser does not need to be differentiable. We apply the feedback-loop mechanism to two examples: generating synthetic genes coding for antimicrobial peptides, and optimizing synthetic genes for the secondary structure of their resulting peptides. A suite of metrics, calculated in silico, demonstrates that the GAN-generated proteins have desirable biophysical properties. The FBGAN architecture can also be used to optimize GAN-generated data points for useful properties in domains beyond genomics.

Synthetic biology refers to the systematic design and engineering of biological systems, and is a growing domain that promises to revolutionize areas such as medicine, environmental treatment and manufacturing[1]. However, current technologies for designing biological products are mostly manual and require significant domain experience. Artificial intelligence can transform the process of designing biological products by helping scientists leverage large existing genomic and proteomic datasets; by uncovering patterns in these datasets, artificial intelligence can help scientists design optimal biological molecules. Generative models, such as generative adversarial networks (GANs), can automate the process of designing DNA sequences, proteins and additional macromolecules for usage in medicine and manufacturing.

Solutions for using GANs for synthetic biology require a framework both for the GAN to generate novel sequences, and also to optimize the generated sequences for desired properties. These properties may include binding affinity of the sequence for a particular ligand, or secondary structure of the generated macromolecule. The possession of such properties by the synthetic molecules may be necessary to enable their real-world use.

Here, we present a novel feedback-loop mechanism for generating DNA sequences using a GAN and then optimizing these sequences for desired properties using a separate predictor, denoted as a function analyser.

The proposed feedback-loop mechanism is applied to train a GAN to generate protein-coding sequences (genes), and then enrich the produced genes for ones coding for antimicrobial peptides (AMPs), and ones coding for α-helical peptides. AMPs are typically lower-molecular-weight peptides with broad antimicrobial activity against bacteria, viruses and fungi[2]. They are an attractive area to apply GANs to since they are commonly short, less than 50 amino acids, and have promising applications in fighting drug-resistant bacteria[3].

Similarly, optimizing for secondary structure is a common task in protein design[4] and is feasible since common secondary structures, such as helices and β-sheets, arise even in short peptides. We optimize for α-helices in particular since these structures are more stable and robust to mutations than β-sheets[5] and AMPs are often helical.

Optimizing for these two properties provides a proof of concept that the proposed feedback-loop architecture FBGAN can be used to effectively optimize a diverse set of properties, regardless of whether a differentiable analyser is available for that property.

Related works

Generative models such as variational autoencoders and recurrent neural networks (RNNs) have also shown promise in producing sequences for synthetic biology applications.

RNNs have shown to be successful in generating SMILES sequences for de novo drug discovery[6] and recent work also showed that RNN outputs could be optimized for specific properties through transfer-learning and fine-tuning on desired sequences[7]. A similar methodology has been applied to generate AMPs[8]. RNNs have been combined with reinforcement learning to produce molecules optimized for biological activity[9].

GANs have the attractive property over RNNs that they allow for latent space interpolation through input codes provided to the generator[10]. GANs are increasingly being used to generate realistic biological data. Recently, GANs have been used to morphologically profile cell images[11], to generate time-series intensive care unit data[12] and to generate single-cell RNA-seq data from multiple cell types[13]. GANs have also been used to generate images of cells imaged by fluorescent microscopy, uniquely using a separable generator where one channel of the image was used as input to generate another channel[14]. The authors of ref. [15] explored using GAN to improve active learning; in contrast, FBGAN uses the feedback from the analyser to improve the GAN process itself.

In independent and concurrent work, Killoran et al. use GANs to generate generic DNA sequences[16]. This work used a popular variant of the GAN known as the Wasserstein GAN, which optimizes the earth mover distance between the generated and real samples[17]. In this approach, the generator was first pretrained to produce DNA sequences, and then the discriminator was replaced with a differentiable analyser. The analyser in this approach was a deep neural network that predicted, for instance, whether the input DNA sequence bound to a particular protein. By backpropagation through the analyser, the authors modified the input noise into the generator

[1]Department of Computer Science, Stanford University, Stanford, CA, USA. [2]Department of Biomedical Data Science, Stanford University, Stanford, CA, USA. *e-mail: jamesz@stanford.edu

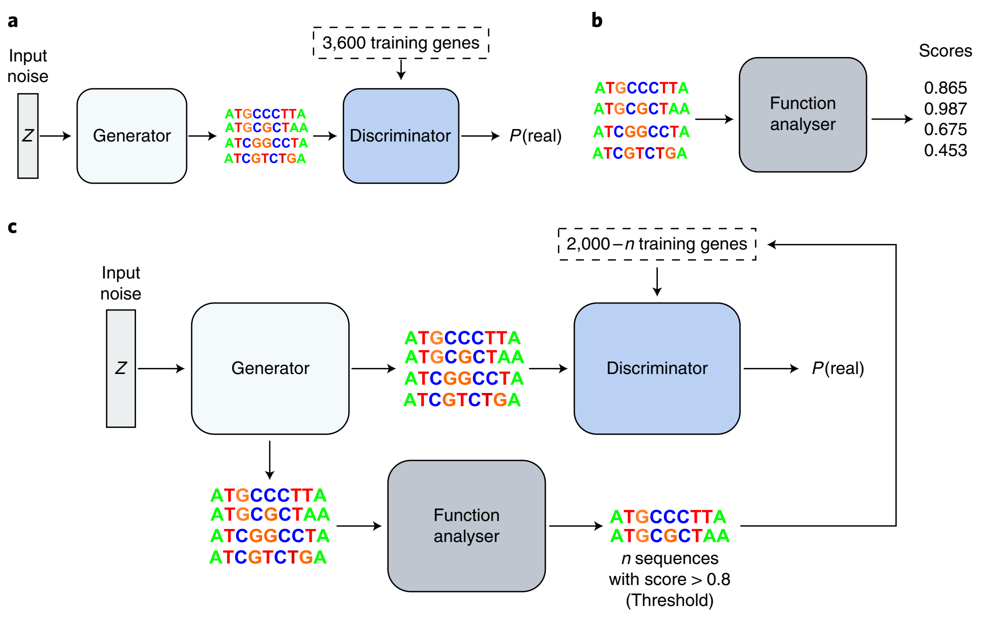

Fig. 1 | FBGAN architecture and training. a, A WGAN pretrained to produce valid genes. **b**, The general form of the function analyser, any black box that takes a sequence and produces a score. **c**, FBGAN's feedback-loop training mechanism. At every epoch, several sequences are sampled from the generator and input into the analyser. The analyser scores each sequence, and sequences above some cutoff are selected to be input back into the discriminator. They replace the least-recently added sequences in the training dataset of the discriminator, so gradually the discriminator's training data are replaced by synthetic data.

to yield desirable DNA sequences. The authors empirically demonstrated that GANs outperformed variational autoencoders for this task of generating DNA sequences. However, the approach of ref. [16] does not extend to non-differentiable analysers, and does not change the generator itself, but rather its input.

Here, we propose a novel feedback-loop architecture, FBGAN, to enrich a GAN's outputs for user-desired properties; an overview of the FBGAN architecture is provided in Fig. 1. FBGAN employs an external predictor to optimize generated data points for the desired property, and this predictor has the added benefit that it does not need to be differentiable. We present a proof of concept of the feedback-loop architecture by first generating realistic genes, or protein-coding DNA sequences, up to 50 amino acids in length (156 nucleotides); feedback is then used to enrich the generator for genes coding for AMPs, and genes coding for α-helical peptides.

Feedback GAN design and training

The basic formulation of a GAN as proposed by Goodfellow et al. consists of two component networks, a generator G and a discriminator D, where the generator G creates new data points from a vector of input noise z, and the discriminator D classifies those data points as real or fake[18]. The end goal of G is to produce data points so realistic that D is unable to classify them as fake. Each pass through the network includes a backpropagation step, where the parameters of G are improved so the generated data points appear more realistic. G and D are playing a minimax game with the loss function in equation (1)[18].

$$\min_{G}\max_{D} V(D,G) = \mathbf{E}_{x\in P_{\text{data}}(x)}[\log(D(x)] + \mathbf{E}_{z\in P(z)}[\log(1-D(G(z))] \quad (1)$$

Concretely, the discriminator seeks to maximize the probability $D(x)$ that x is real when x comes from a distribution of real data, and minimize the probability that the data point is real, $D(G(z))$, when $G(z)$ is the generated data.

The Wasserstein GAN (WGAN) is a variant of the GAN thath instead minimizes the earth mover (Wasserstein) distance between the distribution of real data and the distribution of generated data[17]. A gradient penalty is imposed for gradients above one to maintain a Lipshitz constraint[19].

WGANs have been shown empirically to be more stable during training than the vanilla GAN formulation. Moreover, the Wasserstein distance corresponds well to the quality of the generated data points[17].

Our generative models for producing genes follow the WGAN architecture with the gradient penalty proposed by Gulrajani et al.[19]. The model has five residual layers with two one-dimensional convolutions of size 5×1 each. The softmax in the final layer is replaced with a Gumbel softmax operation with temperature $t=0.75$. When sampling from the generator, the argmax of the probability distribution is taken to output a single nucleotide at each position. Models were coded in Pytorch and initially trained for 70 epochs with a batch size $B=64$.

GAN dataset. We first assembled a training set of naturally observed protein sequences. More than 3,655 proteins were collected from the Uniprot database, where each protein was less than 50 amino acids in length[20]. These proteins were selected from the set of all reviewed proteins in Uniprot with length from 5–50 residues, and the protein sequences were then clustered by sequence similarity ≥0.5. One representative sequence was selected from each cluster to form a diverse dataset of short peptides. The dataset was limited to proteins up to 50 amino acids in length since this length allows for observations of protein properties such as secondary structure and binding activity, while limiting the long-term dependencies the GAN would have to learn to generate protein-coding sequences.

The Uniprot peptides were then converted into complementary DNA sequences by translating each amino acid to a codon (where a random codon was selected when multiple codons mapped to one amino acid); the canonical start codon and a random stop codon were also added to each sequence. All sequences were padded to length 156, which was the maximum possible length.

Feedback-loop training mechanism. The feedback-loop mechanism consists of two components (Fig. 1). The first component is the WGAN, which generates novel gene sequences not yet enriched for any properties. The second component is the analyser; in our first use case, the analyser is a differentiable neural network that takes in a gene and predicts the probability that the gene codes for an AMP. However, the analyser can be any black box that takes in a sequence and assigns a 'favourability' score to the sequence. In our second use case, the analyser is a web server that returns the number of α-helical residues a gene will code for. An advantage of FBGAN is that it can work with any analyser and does not require the analyser to be differentiable.

The GAN and analyser are linked by the feedback mechanism after an initial number of pretraining epochs; pretraining is done so that the generator is producing valid sequences for input into the analyser. Once feedback starts, every epoch several sequences (here 15 batches) are sampled from the generator and input into the analyser. The analyser assigns a score to each sequence, and all sequences with a score above the cutoff are input back into the discriminator's dataset. The generated sequences replace the least-recently added (oldest) genes in the discriminator's training dataset. The generator's ability to mimic these new sequences is thus factored into the loss.

The GAN is then trained as usual for one epoch (one pass through this training set). As the feedback process continues, the entire training set of the discriminator is replaced repeatedly by high-scoring generated sequences.

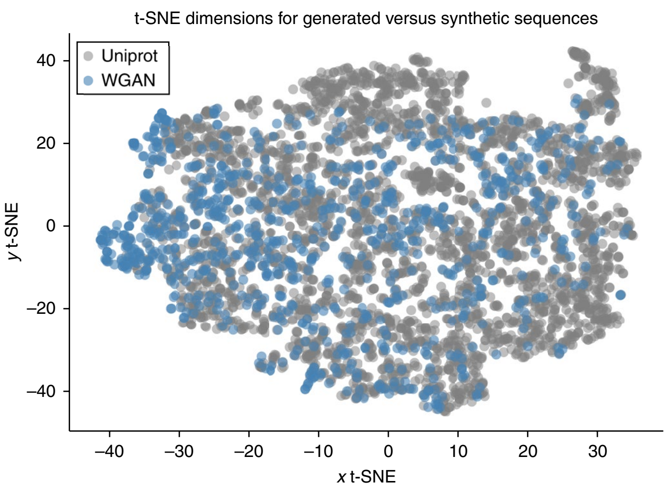

Fig. 2 | t-SNE visualization of synthetic genes. A set of 500 valid synthetic genes were randomly sampled from the trained WGAN, and 10 physiochemical features were calculated for the encoded proteins. The same ten features were also calculated for the cDNA sequences from Uniprot proteins. Dimensionality reduction was conducted through t-SNE with perplexity 40. The synthetic proteins cover a similar data manifold as the Uniprot proteins.

Analyser for AMP-coding genes. The first analyser is a recurrent classifier with gene input and output probability that the gene codes for an AMP.

Dataset. The AMP classifier was trained on 2,600 experimentally verified AMPs from the APD3 database[21], and a negative set of 2,600 randomly extracted peptides from UniProt from 10 to 50 amino acids (filtered for unnatural amino acids). The dataset was loaded using the Modlamp package[22]. As above, the proteins were translated to cDNA by translating each amino acid to a codon (a random codon in the case of redundancy), and by adding a start codon and random stop codon. The AMP training dataset was split into 60% training, 20% validation and 20% test sequences.

Classifier architecture. Using Pytorch, we built and trained a RNN classifier to predict whether a gene sequence would produce an AMP. The architecture of the RNN consisted of two gated recurrent unit (GRU) layers with hidden state $h = 128 \times 1$. The second GRU layer's output at the final time step was fed to a dense output layer, with the number of neurons equal to the number of output classes minus one. This dense layer had a sigmoid activation function, such that the output corresponded to the probability of the gene sequence being in the positive class. To reduce overfitting and improve generalization, we added dropout with $p = 0.3$ in both layers.

Using the Adam optimizer with learning rate $\mathrm{lr} = 0.001$, we optimized the binary cross-entropy loss of this network. The network was trained using minibatch gradient descent with batch size $B = 64$, and 60% of the data were retained for training, 20% for validation and 20% for testing. The model was trained for 30 epochs.

Secondary-structure black-box analyser. To optimize synthetic genes for secondary structure, a wrapper was written around the PSIPRED predictor of secondary structure[23]. The PSIPRED predictor takes in an amino-acid sequence and tags each amino acid in the sequence with known secondary structures, such as α-helix or β-sheet. The wrapper takes in a gene sequence (sampled from the generator), converts it into a protein sequence and predicts the secondary structure of the amino acids in that protein. The wrapper then outputs the total number of α-helix tagged residues. If the gene cannot be converted into a protein (due to a lack of stop/start codon, incorrect length and so on), the wrapper outputs zero. The analyser selects all sequences with helix length above some cutoff to move to the discriminator's training set. In this case, the cutoff was arbitrarily set to five residues.

Results

WGAN architecture to generate protein-coding sequences. Synthetic genes up to 156 nucleotides (50 amino acids) in length are produced from the WGAN; after training, 3 batches of sequences (192) sequences were sampled from the generator. Three batches were also sampled before one epoch of training as a comparison point.

Correct gene structure was defined as a string starting with the canonical start codon 'ATG', followed by an integer number of codons of length 3, and ending with one of three canonical stop codons ('TAA', 'TGA', 'TAG'). Before training, 3.125% of sequences initially followed the correct gene structure. After training, 77.08% of sampled sequences contained the correct gene structure.

To further examine whether the synthetic genes were similar to natural cDNA sequences from Uniprot, physiochemical features such as length, molecular weight, charge, charge density, hydrophobicity and so on were calculated.

t-distributed stochastic neighbour embedding (t-SNE) was conducted on these physiochemical features for the natural and synthetic sequences (Fig. 2). The synthetic proteins occupy a similar data manifold as the Uniprot proteins. In addition, the relative amino-acid frequencies of the synthetic sequences mirror the relative frequencies of the natural cDNA sequences from Uniprot (Supplementary Fig. 2). The WGAN for gene sequences is then used as the generator for the FBGAN architecture. We describe the analyser in the architecture below.

Deep RNN analyser for antimicrobial properties. The AMP analyser is a recurrent classifier that assigns each gene sequence a probability of coding for an AMP. The architecture of the RNN classifier consisted of two GRU layers with hidden state $h = 128 \times 1$

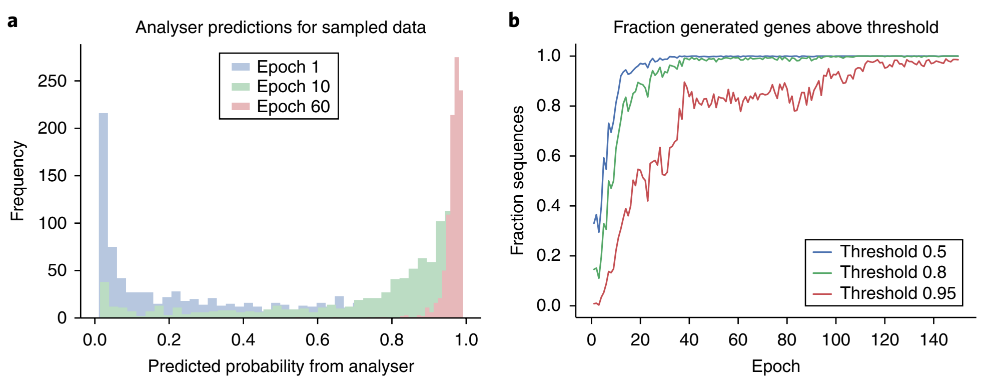

Fig. 3 | AMP analyser predictions over training epochs. a, Histograms showing the predicted probability that generated genes are antimicrobial, as the closed-loop training progresses. While most sequences are initially assigned 0.1 probability of being antimicrobial, as training progresses, nearly all sequences are eventually predicted to be antimicrobial with probability > 0.99. **b**, Percentage of sequences predicted to be antimicrobial with probability above three thresholds: [0.5,0.8,0.99]. While 0.8 was used as the cutoff for feedback, the percentage of sequences above 0.99 also continues to rise during training with feedback.

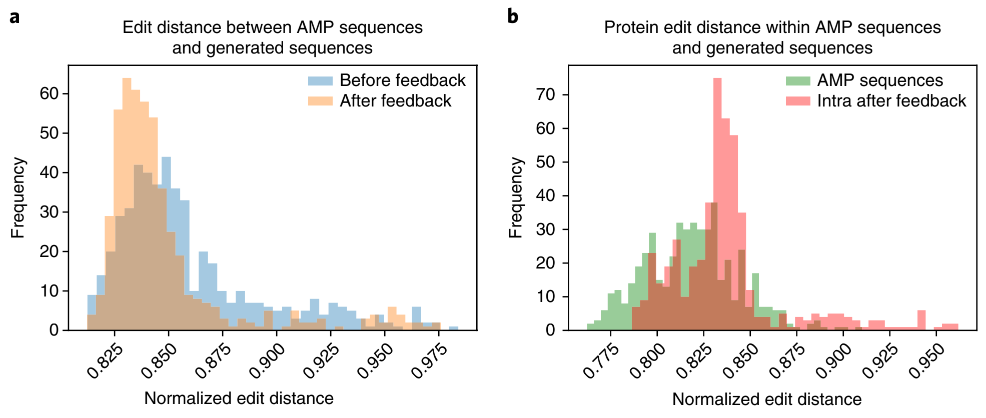

Fig. 4 | Sequence similarity for synthetic AMPs and known AMPs. a, Between-group edit distance (normalized Levenstein distance) between known antimicrobial sequences (AMPs) and proteins coded for by synthetic genes produced without feedback, and between known antimicrobial sequences (AMPs) and proteins coded for by synthetic genes produced after feedback. To calculate between-group edit distance, the distance between each synthetic protein and each AMP was calculated and the means were then plotted. **b**, Within-group edit distance for AMPs and for proteins produced after feedback, to evaluate the variability of GAN-generated genes after the feedback loop. Within-group edit distance was computed by selecting 500 sequences from the group and computing the distance between each sequence and every other sequence in the group; the mean of these distances was then taken and plotted.

Table 1 | Physiochemical properties for known and synthetic AMPs

	Known AMP	**Before feedback**	**After feedback**
	***N*=2,600**	***N*=898**	***N*=18,816**
Length	32.367 ± 0.00692	21.560 ± 0.01484	36.984 ± 0.00090
MW	3,514.007 ± 0.76177	2,427.960 ± 1.65405	4,023.285 ± 0.09831
Charge	3.858 ± 0.00115	2.346 ± 0.00270	2.712 ± 0.00012
ChargeDensity	0.001 ± 0.00000	0.001 ± 0.00000	0.001 ± 0.00000
pI	10.270 ± 0.00079	10.174 ± 0.00274	9.477 ± 0.00010
InstabilityInd	27.174 ± 0.01028	38.255 ± 0.03973	53.156 ± 0.00157
Aromaticity	0.082 ± 0.00002	0.063 ± 0.00008	0.078 ± 0.00000
AliphaticInd	91.859 ± 0.01817	83.751 ± 0.05176	84.890 ± 0.00185
BomanInd	0.770 ± 0.00058	1.812 ± 0.00193	0.886 ± 0.00006
HydrophRatio	0.435 ± 0.00005	0.386 ± 0.00016	0.441 ± 0.00001

Mean ± standard error (sample size given) of physiochemical features, before and after training with feedback. Properties in bold are those for which the mean after feedback is closer to the mean of the known AMPs than before feedback.

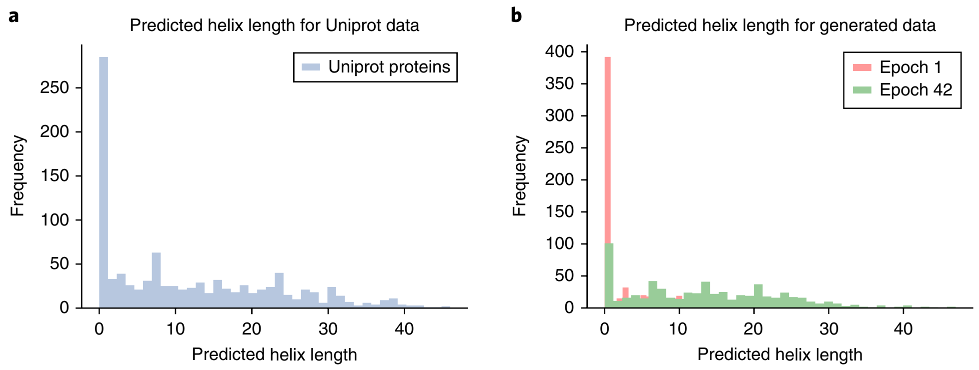

Fig. 5 | α-helix lengths of known versus synthetic proteins. a, Distribution of α-helix lengths for natural proteins under 50 amino acids scraped from Uniprot. **b**, Distribution of α-helix lengths from synthetic gene sequences after 1 and 40 epochs of training. The predicted helix length from the generated sequences quickly shifts to be higher than the helix length of the natural proteins.

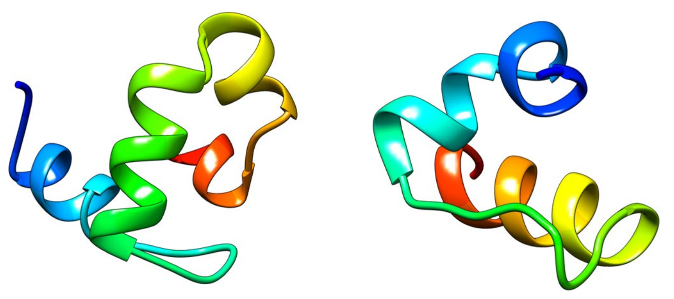

Fig. 6 | Sample α-helices from FBGAN. Example peptides from the synthetic genes output by our WGAN model with feedback from the PSIPRED analyser. Both proteins show a clear helix structure. The peptide on the left was predicted to have 10 residues arranged in helices, while the peptide on the right was predicted to have 22 residues in helices; accordingly, the peptide on the right appears to have more residues arranged in helices.

and dropout $p = 0.3$ in both layers. The function analyser achieved a training accuracy of 0.9447 and a validation accuracy of 0.8613. The test accuracy was 0.842, and the area under the receiver operating characteristic curve on the test set was 0.908. The precision and recall on the test set were 0.826 and 0.8608, respectively, and the area under the precision-recall curve was 0.88 (Supplementary Fig. 3).

Feedback loop to optimize antimicrobial properties. After both the WGAN and function analyser were trained, the two were linked with the feedback-loop mechanism; at each epoch of training, sequences were sampled from the generator and fed into the analyser. The analyser then assigned each sequence a probability of being antimicrobial, and the sequences crossing a user-defined threshold (here with $P(Antimicrobial) > 0.8$) were fed into the discriminator and labelled as 'real' sequences. The n top ranking sequences took the place of the n oldest sequences in the discriminators data set.

Evaluation of FBGAN for AMPs. FBGAN was trained with the feedback mechanism for 150 epochs (until the WGAN distance converged), and almost 20,000 sequences were sampled. The performance of FBGAN was evaluated on several criteria.

The first criterion was whether the percentage of predicted AMPs increased with feedback (without sacrificing gene structure). We examine the analyser's predictions as the training with feedback progresses. As shown in Fig. 3, after only ten epochs of closed-loop training, the analyser predicts the majority of sequences as being antimicrobial. After 60 epochs, nearly all of the sequences are predicted to be antimicrobial with high probability (greater than 0.95).

The threshold for feedback is at 0.8; however, the generator continues to improve beyond the threshold, suggesting that the closed-loop training is robust to changes in threshold. A total of 93.3% of the generated sequences after closed-loop training have the correct gene structure, showing that the reading-frame structure was not sacrificed but rather reinforced.

We also evaluated the FBGAN-generated sequences on an independent AMP predictor to measure whether the feedback loop increased the number of valid AMPs in the generated proteins. Since FBGAN as implemented here relied only on the AMP predictions of a long short-term memory (LSTM) classifier for feedback, we used the open-access AMP prediction tool from the CAMP server as a second, orthogonal test[24]. Among the sequences produced by the WGAN without feedback, 22.2% of the sequences were predicted by CAMP to be valid AMPs. If the WGAN was trained directly on positive AMP sequences without feedback, 31.9% of the sequences are predicted to be valid AMPs. The relatively low agreement could due to discrepancies between the APD3 database of antimicrobial proteins and the CAMP classifier. The outputs of FBGAN saw a significant enrichment, 40.2%, of valid CAMP AMPs.

As an additional baseline comparison, we randomly sampled 3-mers of amino acids from known AMPs and assembled them into new proteins. Only 20.2% of these proteins were predicted by CAMP to be valid AMPs.

Evaluation of physiochemical properties. The generated genes were examined for sequence-level and physiochemical-level similarity to known antimicrobial genes. Figure 4a shows a histogram of the mean edit distance between the known AMPs and proteins from synthetic genes before feedback, and the distance between the AMPs and proteins from synthetic genes produced after feedback.

Figure 4b shows the intrinsic edit distance within the AMP proteins, and within the proteins coded for by the synthetic gene sequences after feedback. All edit distances were normalized by the length of the sequences, in order to not penalize longer sequences unfairly.

The distribution shifts after feedback towards a lower edit distance from the AMP sequences. The sequences after feedback also have a higher edit distance within themselves than the AMP sequences do with each other; this demonstrates that the model has not reached the failure mode of replicating a single data point.

The physiochemical properties of the resulting proteins were measured, and are shown in Table 1. As can be seen in the table, the proteins produced with analyser feedback shift to be closer to the positive AMPs in six out of ten physiochemical properties such as length, hydrophobicity and aromaticity. The proteins produced after feedback remain as similar as proteins produced without feedback for properties like charge and aliphatic index. Violin plots demonstrating shifts for selected properties are shown in Supplementary Fig. 1, and a principal component analysis of physiochemical properties is visualized in Supplementary Fig. 4. The shift occurs even though the analyser operates directly on the gene sequence rather than physiochemical properties, and so the feedback mechanism does not directly optimize for the physiochemical properties that demonstrate a shift.

Optimizing secondary structure with black-box PSIPRED analyser. FBGAN was then applied to produce synthetic genes coding for peptides with a particular secondary structure, here, α-helical peptides. Secondary structure is attractive to optimize since it arises and can be predicted even in short peptides, and is an extremely important property for determining protein function.

The analyser used to optimize for helical peptides is a black-box secondary-structure predictor from the PSIPRED server, which tags protein sequences with predicted secondary structure at each amino acid[23]. All gene sequences with more than five α-helical residues were input back into the discriminator as described in the feedback mechanism.

Evaluation of FBGAN for α-helices. After 43 epochs of feedback, the helix length in the generated sequences was significantly higher than the helix length without feedback and the helix length of the original Uniprot proteins, as illustrated by Fig. 5.

In addition, we independently folded several of the peptides to verify that overfitting was not occurring. Folded examples of peptides generated are shown in Fig. 6; these three-dimensional peptide structures were produced by ab initio folding from our generated gene sequences, using knowledge-based force-field template-free folding from the QUARK server[25]. The edit distance within the DNA sequences generated after feedback was in the same range as the edit distance within the Uniprot natural cDNA sequences (Supplementary Fig. 5), and higher than the edit distance within the sequences from WGAN with no feedback.

As a baseline comparison to evaluate the difficulty of generating α-helices, we sampled 3-mer residues from actual protein sequences with known amphipatic helices. We then combined these 3-mers together into synthetic peptides. These sequences had an average of only 1.42 helical residues, well below the length of helices generated by FBGAN.

Discussion

In this work, we have developed the FBGAN architecture to produce protein-coding sequences for peptides under 50 amino acids in length, and demonstrated a novel feedback-loop mechanism to optimize those sequences for desired properties. We use a function analyser to assign a score to sampled sequences from the generator at every epoch, and input sequences above some threshold back into the discriminator as 'real' data points. In this way, the outputs from the generator slowly shift over time to outputs that are assigned a high score by the function analyser. We have shown that the feedback-loop training mechanism is robust to the type of analyser used; as shown in the secondary-structure case, the analyser does not have to be differentiable for the feedback mechanism to be successful.

We have demonstrated the usefulness of the feedback-loop mechanism in two use cases: optimizing for genes that code for AMPs, and optimizing for genes that code for α-helical peptides. For the first use case, we built our own deep RNN analyser as well as evaluating on an independent AMP predictor; for the second, we employed the existing PSIPRED analyser in a black-box manner. In both cases, we were able to significantly shift the generator to produce genes likely to have the desired properties.

FBGAN produced a higher proportion of valid gene sequences that were predicted to be AMPs, when compared to both a kmer baseline and a WGAN trained directly on known AMPs.

Being able to optimize synthetic data for desired properties without a differentiable analyser is useful for two reasons. The first is that the user might not have enough positive data to effectively train a differentiable classifier to distinguish positive data points. In FBGAN, the analyser can be any model that takes in a data point and assigns it a score; the analyser may now even be a machine carrying out experiments in a laboratory.

The second reason is that many existing models in bioinformatics are based on non-differentiable operations, such as BLAST searches or homology detection algorithms. This feedback-loop mechanism allows such models to integrate smoothly with GANs.

While we were able to extend the GAN architecture to produce genes up to 156 base pairs in length while maintaining the correct start codon/stop codon structure, it was noticeably more difficult to maintain the structure at 156 base pairs than at 30 base pairs or 50. To allow the generator to learn patterns in the sequence over longer lengths, we might investigate using a recurrent architecture or even dilated convolutions in the generator, which have been shown to be effective in identifying long-term genomic dependencies[26]. It is still challenging to use GAN architectures to produce long, complex sequences, which currently limits the usefulness of GANs in designing whole proteins, which can be thousands of amino acids long.

Here, to make the training process for the GAN easier, we focus on producing gene sequences that have a clear start/stop codon structure and only four nucleotides in the vocabulary. However, in the future, we might focus on producing protein sequences directly (with a vocabulary of 26 amino acids).

While we have shown that the proteins from the synthetic genes have shifted after training to be more physiochemically similar to known AMPs, we would like to conduct additional experimental validation on the generated peptides. The same holds true for the predicted α-helical peptides.

In future work, we would also like to apply and further validate the currently proposed method on additional application areas in genomics and personalized medicine, such as noncoding DNA and RNA. In addition, FBGAN's proposed feedback-loop mechanism for training GANs is not limited to sequences or to synthetic biology applications; thus, our future work also includes applying this methodology to non-biological use cases of GANs.

Data availability

Demo, instructions and code for FBGAN are available at https://github.com/av1659/fbgan. All of the data used in this paper are publicly available and can be accessed at the references cited[22].

Received: 20 June 2018; Accepted: 4 January 2019;
Published online: 11 February 2019

References

1. Benner, S. & Sismour, M. Synthetic biology. *Nat. Rev. Genet.* **6**, 533–543 (2005).
2. Izadpanah, A. & Gallo, R. Antimicrobial peptides. *J. Am. Acad. Dermatol.* **52**, 381–390 (2005).
3. Papagianni, M. Ribosomally synthesized peptides with antimicrobial properties: biosynthesis, structure, function, and applications. *Biotechnol. Adv.* **21**, 465–499 (2003).
4. Rocklin, G. J. et al. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science* **357**, 168–175 (2017).
5. Abrusán, G. & Marsh, J. Alpha helices are more robust to mutations than beta strands. *PLoS Comput. Biol.* **12**, e1005242 (2016).
6. Segler, M., Kogej, T., Tyrchan, C. & Waller, M. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS Cent. Sci.* **4**, 120–131 (2018).

7. Gupta, A. et al. Generative recurrent networks for de novo drug design. *Mol. Inform.* **37**, 1700111 (2018).
8. Muller, A. T., Hiss, J. A. & Schneider, G. Recurrent neural network model for constructive peptide design. *J. Chem. Inf. Model.* **58**, 472–479 (2018).
9. Olivecrona, M., Blaschke, T., Engkvist, O. & Chen, H. Molecular de novo design through deep reinforcement learning. *J. Chemoinformatics* **9**, 48 (2017).
10. Salimans, T. et al. Improved techniques for training GANs. Preprint at abs/1606.03498 (2016).
11. Goldsborough, P., Pawlowski, N., Caicedo, J., Singh, S. & Carpenter, A. Cytogan: generative modeling of cell images. Preprint at https://www.biorxiv.org/content/10.1101/227645v1 (2017).
12. Esteban, C., Hyland, S. & Rätsch, G. Real-valued (medical) time series generation with recurrent conditional GANs. Preprint at https://arxiv.org/abs/1706.02633 (2017).
13. Ghahramani, A., Watt, F. & Luscombe, N. Generative adversarial networks uncover epidermal regulators and predict single cell perturbations. Preprint at https://www.biorxiv.org/content/10.1101/262501v2 (2018).
14. Osokin, A., Chessel, A., Carazo-Salas, R. & Vaggi, F. GANs for biological image synthesis. Preprint at http://arxiv.org/abs/1708.04692 (2017).
15. Zhu, J. & Bento, J. Generative adversarial active learning. Preprint at https://arxiv.org/abs/1702.07956 (2017).
16. Killoran, N., Lee, L., Delong, A., Duvenaud, D. & Frey, B. Generating and designing DNA with deep generative models. Preprint at https://arxiv.org/abs/1712.06148 (2017).
17. Arjovsky, M., Chintala, S. & Bottou, L. Wasserstein generative adversarial networks. In *Proc. 34th International Conference on Machine Learning* (eds Precup, D. & Teh, Y. W.) 214–223 (PMLR, 2017).
18. Goodfellow, I. et al. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27* (eds Ghahramani, Z. et al.) 2672–2680 (Curran Associates, 2014).
19. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. & Courville, A. Improved training of Wasserstein GANs. Preprint at https://arxiv.org/abs/1704.00028 (2017).
20. Apweiler, R. et al. Uniprot: the universal protein knowledgebase. *Nucleic Acids Res.* **46**, 2699 (2018).
21. Wang, G., Li, X. & Wang, Z. Apd3: the antimicrobial peptide database as a tool for research and education. *Nucleic Acids Res.* **44**, D1087–D1093 (2016).
22. Muller, A., Gabernet, G., Hiss, J. & Schneider, G. modlamp: python for antimicrobial peptides. *Bioinformatics* **33**, 2753–2755 (2017).
23. Buchan, D., Minneci, F., Nugent, T., Bryson, K. & Jones, D. Scalable web services for the PSIPRED protein analysis workbench. *Nucleic Acids Res.* **41**, W349–W357 (2013).
24. Waghu, F., Barai, R., Gurung, P. & Idicula-Thomas, S. Campr3: a database on sequences, structures and signatures of antimicrobial peptides. *Nucleic Acids Res.* **44**, D1094–D1097 (2016).
25. Xu, D. & Zhang, Y. Ab initio protein structure assembly using continuous structure fragments and optimized knowledge based force field. *Proteins* **80**, 1715–1735 (2012).
26. Gupta, A. & Rush, A. Dilated convolutions for modeling long-distance genomic dependencies. Preprint at https://arxiv.org/abs/1710.01278 (2017).

Acknowledgements
The authors thank A. Kundaje for guidance when initiating the research on GANs and DNA. J.Z. is supported by a Chan-Zuckerberg Biohub Investigator grant and National Science Foundation (NSF) grant CRII 1657155.

Author contributions
J.Z. conceived the objective of using GANs to generate genes and optimize protein functions; A.G. conceived of and implemented the feedback-loop architecture and conducted the experiments and analysis. Both authors wrote the manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary information is available for this paper at https://doi.org/10.1038/s42256-019-0017-4.

Reprints and permissions information is available at www.nature.com/reprints.

Correspondence and requests for materials should be addressed to J.Z.

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.