Learning to Generate 5' UTR Sequences for Optimized Ribosome Load and Gene Expression

Sina Barazandeh[1] Furkan Ozden[1,2] Ahmet Hincer[3]
Urartu Ozgur Safak Seker[3,*] A. Ercument Cicek[1,4,*]
[1]Department of Computer Engineering, Bilkent University, Ankara, Turkey
[2]Department of Computer Science, Oxford University, Oxford, UK
[3]National Nanotechnology Research Center, Bilkent University, Ankara, Turkey
[4] Computational Biology Department, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

The 5' untranslated region (5' UTR) of the messenger RNA plays a crucial role in the translatability and stability of a molecule. Thus, it is an important component in the design of synthetic biological circuits for high and stable expression of intermediate proteins. Several UTR sequences are patented and used frequently in laboratories. We present a novel model UTRGAN, a Generative Adversarial Network (GAN)-based model designed to generate 5' UTR sequences coupled with an optimization procedure to ensure a target property such as high expression for a target gene sequence or high ribosome load. We rigorously analyze and show that the model can generate sequences that mimic various properties of natural UTR sequences. Then, we show that the optimization procedure yields sequences that are expected to yield 32% higher expression (up to 7-fold) on a set of target genes and 12% higher ribosome load on average on a set of generated 5' UTRs (up to 90% for the best 5' UTR), compared to the initially generated UTR sequences. We also demonstrate that when there is a single target gene of interest, the expected expression increases by 55% on average and up to 100% for certain genes (up to 15-fold for the best 5' UTR).

1 Introduction

Controlling gene expression has many applications in synthetic biology and RNA-based therapeutics [1], and the stability of the RNA plays a vital role in the control. Thus, designing stable and highly-expressed RNA sequences is desirable [2][3][4]. The 5'-untranslated region (5' UTR) of a gene is an important component, as it affects the mRNA level as well as the stability of the produced transcript [5]. Although the 5' UTR of the mRNA is not translated to protein, it is also crucial in regulating the translation process because it contains the ribosome binding site (RBS) in which the ribosome attaches to initiate the translation. For this reason, many UTR sequences have been patented and used in the design of gene circuits [6][7].

The standard approach to optimize UTR sequences is by introducing variants in existing target UTR and evaluating the effectiveness by using simple algorithmic approaches. Von Niessen *et al.* [7] construct a library of 5' UTR sequences based on natural samples that are extracted from the human genome and generate synthetic sequences using genetic algorithms. They also use scoring functions based on the characteristics of the sequences, such as GC content, k-mer frequency, and

*Co-corresponding Authors. Emails: cicek@cs.bilkent.edu.tr and urartu@bilkent.edu.tr

RNA folding energy. The genetic algorithm here is used to avoid experimenting with the entire space of possible sequences. The sequences are then selected according to their predicted translation efficiency [6]. Similarly, another study identifies preferable 3' UTR sequences in providing binding sites for regulatory proteins [8]. There are also studies identifying both 3' and 5' UTRs for higher gene expression, and the samples are selected from naturally abundant mRNA sequences in human tissues [9]. These models use machine learning algorithms only to predict sequence features and use those features to evaluate the generated samples. Nevertheless, machine learning can be further used for other tasks, such as generating novel data samples based on existing biological samples [10].

Several studies employ generative adversarial networks (GANs) to generate sequences with similar characteristics to the actual biological DNA samples. Zirmec et al. propose a framework based on GANs for generating 1,000 bps yeast DNA sequences [11][12]. They employ the model proposed by Zrimec *et al.* [13] that uses the regulatory regions to predict the yeast gene expression and utilize this model to optimize the gene expression by updating the entire DNA sequence. However, the model is suitable only for bacteria because the DNA sequences are short. Thus, this model generates complete DNA sequences and optimizes the entire sequence, and therefore is not capable of generating the regulatory sequences separately that can be plugged into other target genes of interest. To the best of our knowledge, there is no prior work that is tailored for generating and optimizing regulatory regions.

In this work, we propose UTRGAN, the first GAN-based pipeline for generating novel human 5' UTR sequences and optimizing them for higher Mean Ribosome Load (MRL) and higher gene expression. Our model generates 5' UTR sequences that can be attached to any real or synthetic gene of interest. UTRGAN can generate 5' UTR sequences without modifying the target DNA/RNA sequence it is attached to. Our generated sequences resemble the natural 5' UTRs in several important characteristics, such as GC content distribution, k-mer distance, and Minimum Free Energy (MFE) [14]. Furthermore, the optimization procedure enables the generation of optimized sequences for higher MRL and mRNA abundance. By using multiple sequences for optimization, we increase the aforementioned metrics by 12% and by 32% compared to the initial values, respectively. Depending on the application, our model is able to optimize the generated 5' UTRs on a specific target DNA sequence or on a set of DNA sequences. The optimization for a single gene of interest increases the expression by 55% on average and can result in up to 28 times higher expression for the best synthetic 5' UTR. UTRGAN's ability to generate and optimize 5' UTRs for any given gene sequence will be the key enabler for the genetic circuit design. We believe that UTRGAN will pave the way for mRNA-based therapeutics in biotech industry for any application that requires gene expression control, such as in cancer immunotherapy [15].

2 Results

2.1 Overview of UTRGAN

The UTRGAN model is a deep generative adversarial network (GAN) that learns generating 5' UTR sequences with similar characteristics to the natural ones. This model is a novel version of the WGAN-GP architecture [16] for one-hot encoded DNA sequence. The model provides improved training over the original Convolutional GAN (DCGAN) models [16][17] and is less prone to overfitting than the original WGAN architecture [18].

Figure 1A shows the overview of the architecture. The input is a noise vector. The generator and the critic are convolutional neural networks trained together. The generator upsamples the input vector using transpose-convolutions to generate 5' UTR sequences, whereas the critic uses

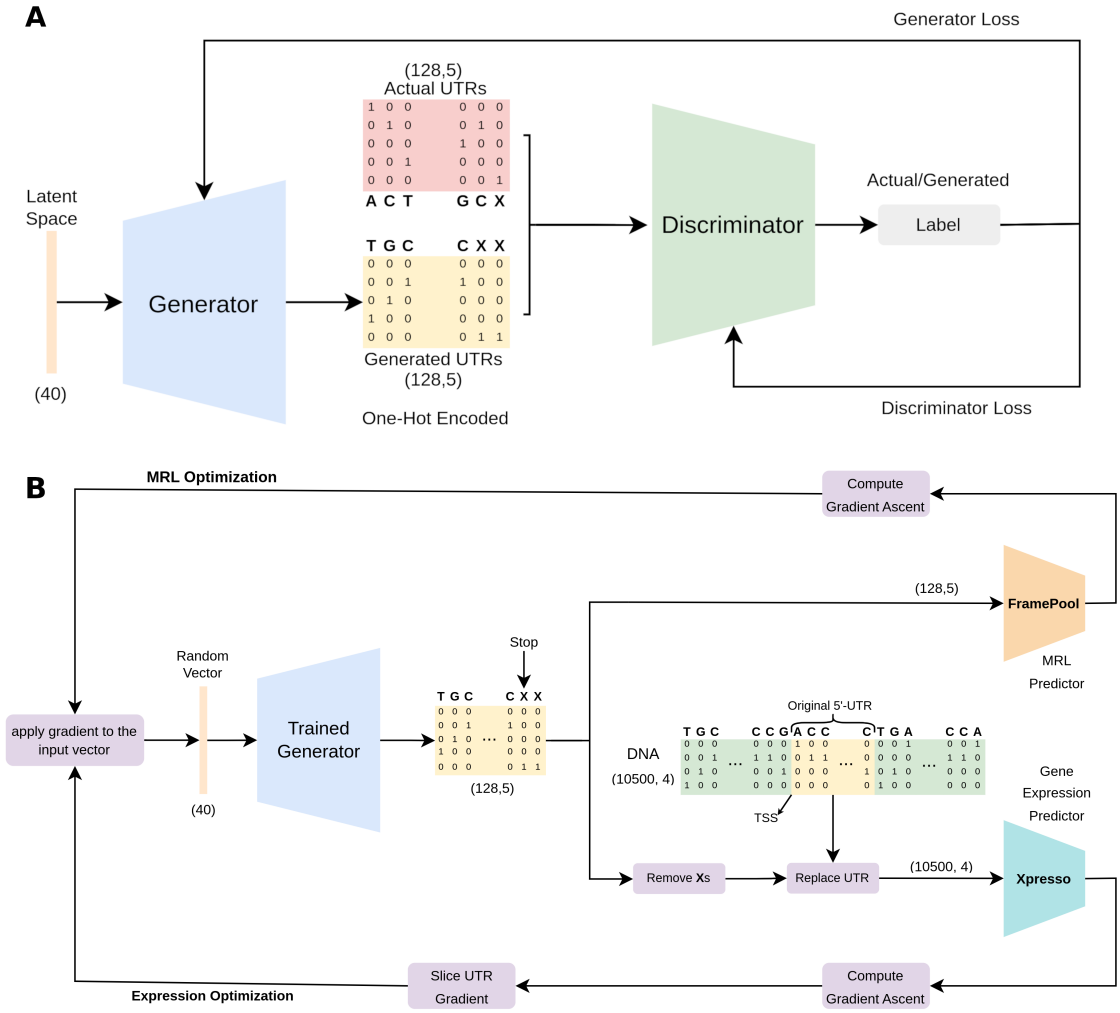

Figure 1: Shown on **A** is the GAN training phase of the GAN model where the competition between the Discriminator and the Generator and updating their weights based on the types of samples is used to train the models. The diagram of the pipeline shown on **B** is where the optimization takes place. The GAN is used to generate samples, and the input noise is updated using the Gradient Ascent algorithm to increase the predicted MRL or mRNA abundance of the generated 5' UTR sequences. These two paths with different considerations are for MRL and mRNA optimization. The mRNA optimization requires attaching the 5' UTR to a DNA and then slicing the relevant gradient to update the input noise, whereas the MRL gets 5' UTRs as its input. Therefore, the optimization is more straightforward for MRL compared to mRNA abundance.

convolutions and dense layers to score the actual and generated samples. The optimization pipeline is shown in Figure 1B where either one of the two predictor models is used [19][20] for predicting the gene expression or MRL values. We perform optimization by updating the initial input vector until the model converges. That is, we apply Gradient Ascent on the generator's input, as the target is to maximize the MRL or gene expression values. We use off-the-shelf deep-learning models that predict gene expression [19] and MRL [20] given a sequence for the optimization procedure.

2.2 The model generates novel 5' UTR sequences

We use tSNE and UMAP to map 2000 generated and 120 natural human 5' UTRs with a randomly selected length of 78 to compare the embeddings of the sequences. We also generate 200 random sequences with the same length using the DNA alphabet. This fixed length is necessary since the tSNE and UMAP require all the samples to be of the same dimensions.

The plots shown in Figure 2 indicate two points about the generated samples. First, the distribution of the sequences in blue shows that the generated 5' UTRs are mostly not similar to any of the random or actual sequences. That is, the model is able to generate structurally novel sequences that are different from both actual and random sequences. However, generated samples are also not a distinct cluster. This is shown in the literature both in the visual and biological domain that the generated samples in a generative model are expected to have a certain level of resemblance and a reasonable distance [21][13].

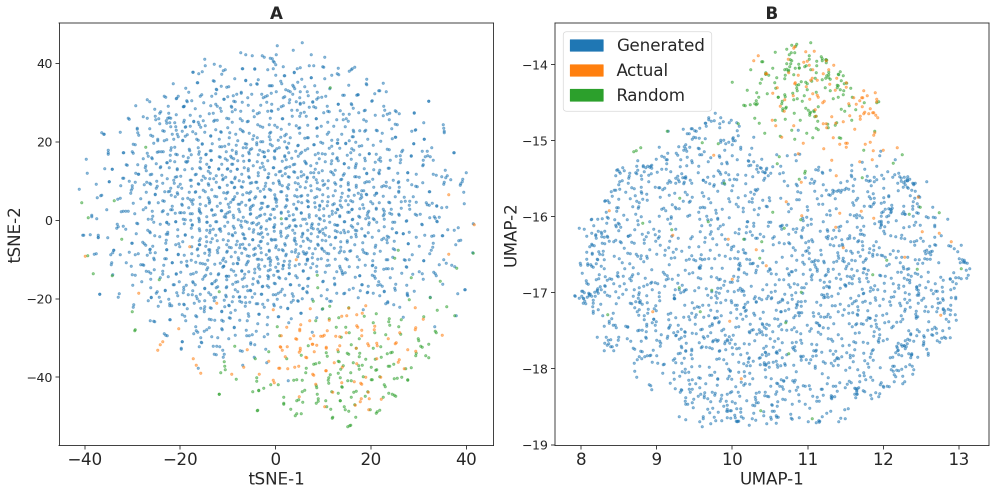

Figure 2: The visualization of the generated, actual, and random sequences using tSNE and UMAP dimension reduction methods show that the generated samples are different from the original 5'-UTR sequences while many of their characteristics such as GC-Content distribution, minimum 4-mer distance, Mean-Ribosome-Load distribution, Minimum-Free-Energy distribution resemble the actual samples. As shown in the plot, the random samples are closer to the actual ones in lower dimensions, while their characteristics are not similar to the distribution of the actual samples.

2.3 4-mer distributions are similar in actual and generated UTRs

Another metric to measure the similarity among two sets of sequences is to use the distribution of the 4-mers in the sequences as also done in the literature [22]. We compute the frequency of the 4-mers for each sequence. Then, for each sequence in the random, actual and generated set, we find the closest sequence (Squared Euclidean) in the entire set of actual 5' UTRs, ignoring identical sequences (so actual sequences match to the nearest neighbor). We obtain a distribution per group. We observe very similar distance distributions for the generated and actual samples as shown in Figure 3A, indicating that the generated set of sequences retain structural similarity to natural sequences and yet, are distinct in terms of the sequence itself, as shown in the previous subsection. In accordance, we observe that the random sequences have a different 4-mer distribution despite being similar to natural sequences in terms of the sequence itself (see Figure 2).

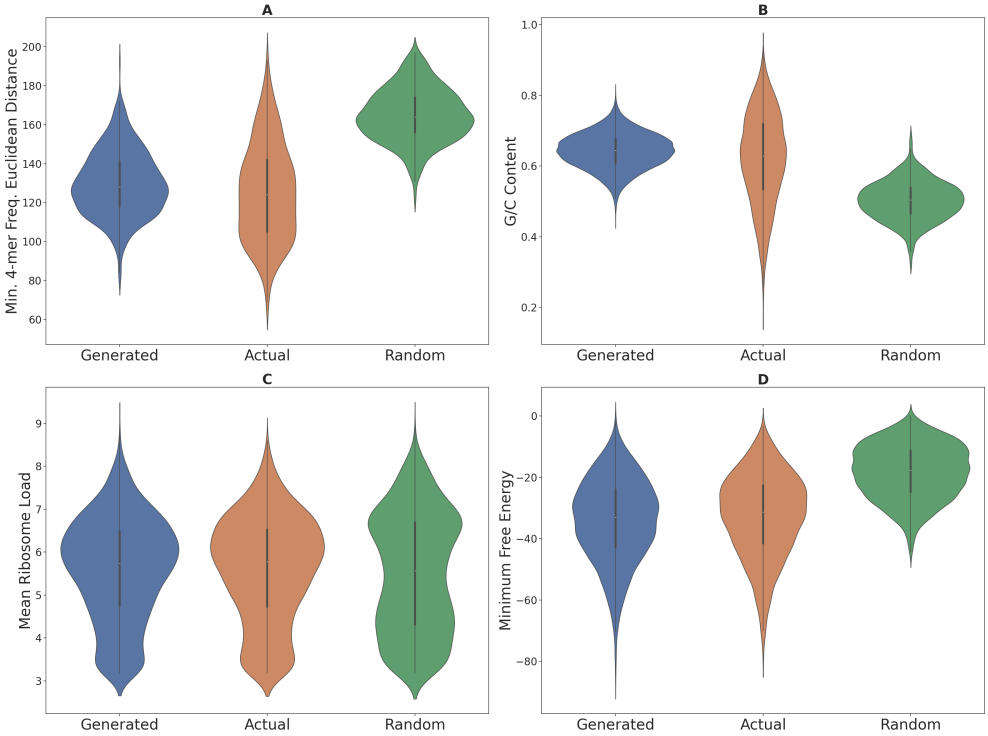

Figure 3: **Comparison of actual, generated, and random sequences. A**) We calculate the 4-mer frequencies in each sequence and select the distance to the closest non-identical actual sample; The distribution of these values in generated sequences is more similar to the actual sequences, and the average minimum distance of the random sequences is higher showing that the random ones have different 4-mer frequency distributions. **B**) The distribution of the GC content of the random sequences is expected to be centered at 50 percent due to the nature of random sequences. This distribution for the generated sequences, on the other hand, is aligned with the actual ones. However, the distribution is still not as wide as that of the actual ones and does not include very high or low GC content sequences. The GC content of the generated sequences is limited to the 50 to 70 percent range. **C**) The distribution of the mean ribosome load (MRL) in the generated and actual sequences is almost identical, whereas the random ones, although they cover the same range, are distributed differently. Random sequences are also in variable lengths like the generated and actual to eliminate the possible effect of fixed length on the MRL. **D**) The distribution of minimum free energy (MFE) of the generated sequences resembles that of actual sequences. The random ones are distributed considerably differently and cover a smaller range of MFE values.

2.4 GC content distribution is consistent in the generated 5'-UTRs

Another measure is the percentage of guanine-cytosine content (GC content) in a sequence that is informative regarding the stability of the sequences [23]. We expect the distribution of the GC content in the natural and generated sequences to be similar, and we use 10,000 random, 10,000 generated, and 9,280 actual sequences in this analysis. The same number of sequences is used for the experiments in the following subsections except if stated otherwise. We observe that the mean GC content in the generated and actual samples are very similar, and the generated samples have a smaller variance (Figure 3B). This is in contrast to the GC content for random sequences with

the length range that is centered at 50%, as expected.

2.5 Mean Ribosome Load of the natural and generated have similar distributions

While the above-mentioned metrics are informative about the performance of the model in generating samples structurally similar to the actual sequences, they do not take into account any specific characteristic of the sequence in terms of biology. Mean ribosome load (MRL) is a metric defined based on the ribosome count associated with an mRNA and is considered a proxy for translation rate [20]. To measure the similarity of the sequences from a biological perspective, we use MRL, in which load is the average number of ribosomes associated with a given RNA as a proxy for translation efficiency. To estimate the MRL of a given 5' UTR sequence, we use a convolutional neural network model (FramePool) that predicts the MRL of a 5' UTR [20] with any length. This model utilizes a Frame-Pooling layer which processes indices of the frames of three consecutive nucleotides (codons) separately. Figure 3C shows that the MRL distributions of the actual and the generated sequences are almost identical. In contrast, the random sequences are more likely to generate lower MRL values compared to the former, as expected.

2.6 Generated and actual samples have similar Minimum Free Energy distributions

The minimum free energy (MFE) of a 5' UTR sequence as part of the RNA sequence is another informative characteristic of the sequence regarding its stability. It depends on the number, composition, and arrangement of the nucleotides in the mRNA sequence [14] and the 5' UTR in our case. Similar to the GC content, even though MFE might not be the sole indicator of any characteristic of a given 5' UTR sequence, we expect to see similar distributions of MFE values in a large number of samples. As shown in Figure 3D, the distribution of this value is very similar in generated and natural sequences while very different in the set of randomly generated sequences.

2.7 Optimization yields sequences with higher expected expression

The optimization procedure, as explained in Section 4, is based on an iterative procedure that updates the input noise of the GAN to generate 5' UTR sequences with higher expression. We use the model to optimize the predicted expression of the generated sequences. The model used for prediction is a convolutional neural network-based model (Xpresso [19]) that outputs the predicted log TPM expression for the input DNA sequence. As shown in Figure 4A, the optimization of 256 generated 5' UTR sequences for maximizing the average expression of a set of 100 randomly selected human genes results in increased expression in 80% of the DNA samples. To calculate a single score for each 5' UTR, we average the expression of all gene sequences with the 5' UTR attached to them. Note that we are limited by updating the 5' UTR only, and 5' UTR is not the sole determinant of the expression. Yet, we observe an increase in expression that reach up to 90% compared to the initial value. In 20% of the 5' UTRs, the optimization results in reduced expression compared to the initial generated sequence. In those cases, the user can pick the initially generated 5' UTR. We discuss the possible reasons for degradation in Section 3.

2.8 Mean Ribosome Load increases after optimization

The FramePool [20] model used to predict MRL can predict the value for sequences of any length and does not require the 5' UTR to be attached to an mRNA. Using this model and optimizing the

initially generated 5' UTRs in 1,000 iterations, the results show increased MRL in more than 70% of the generated 5' UTRs (See Figure 4B). In this experiment, we optimize 128 generated 5' UTR sequences and select 100 of them randomly for visualization. Unlike the mRNA abundance, the MRL is not a DNA-specific metric and does not require a target DNA or mRNA for optimization. To combine the optimization for mRNA abundance and MRL, one can optimize many sequences for gene expression first and then select the ones also with higher MRL.

2.9 Optimization improves the expression for specific target genes

As discussed above, our optimization procedure increases the gene expression when we optimize multiple generated 5' UTRs for multiple gene sequences (optimization based on average). Instead, we can work with a single target gene as well. To optimize 5' UTRs for a specific gene, we generate 128 5' UTR sequences using a fixed seed and optimize them for higher expressions when attached to the target gene only. We optimize 5' UTRs for the *TLR6*, *TNF*, *IFNG*, and the *TP53* genes from the human genome. We show in Figure 5 that optimization for these genes improves gene expression for more than 95% of the generated 5' UTRs. It provides 90%, 100%, 35%, and 29% increase in expression on average for each gene, respectively. While these results show that optimization is successful on average, users can pick only the top generated 5' UTR for their application. The performance increase for the highest expression yielding 5'-UTR is 7-fold, 4-fold, 28-fold, and 3.5-fold for each respective gene. The best 5' UTR sequences for these genes are shown in Supplementary Table 1.

Noreen and Arshad [24] discuss the role of the *TLR6* gene in the regulation of innate as well as adaptive immunity. Based on its role in the immune system, the expression level of this gene can affect the immune response. Another gene optimized here that has a role in the immune system is the *IFNG* (*IFN-*γ) gene [25][26]. The other selected gene is *TNF* or *TNF-*α, which is believed to have pathogenetic and therapeutic applications [27]. The last gene we optimize is the *TP53* gene, which is a tumor-related gene and a cancer suppressor [28][29]. Controlling the expression of these genes can be of interest based on their applications. These are provided as examples to show the capacity of the model to increase the expression of specific target genes, and this approach can be used on any gene for expression optimization.

2.10 Replacing natural 5' UTRs for target genes

Our main goal in this study is not to replace existing 5' UTRs of natural genes. The experiments in the previous subsections with a single or a set of target genes are to show that the optimization mechanism can tailor the initially generated 5' UTR sequences toward a goal (maximize expression) using natural gene sequences as examples. This application is rather intended for DNA molecules which are expected to be novel and synthetic with some desired properties such as binding to another protein after translation. Yet, we investigate if the 5' UTRs we generate yield better expected expression compared to the natural 5' UTRs. For each of the *TLR6*, *IFNG*, *TNF*, and *TP53* genes, we generate 128 5'-UTR sequences and pick the sequence that maximizes the expected expression and compare this value with the expected expression of the gene using the original natural 5' UTR. We observe that the sequences we generate yield 28-fold, 7-fold, 4-fold, and 3.5-fold increases in the expected expression, respectively.

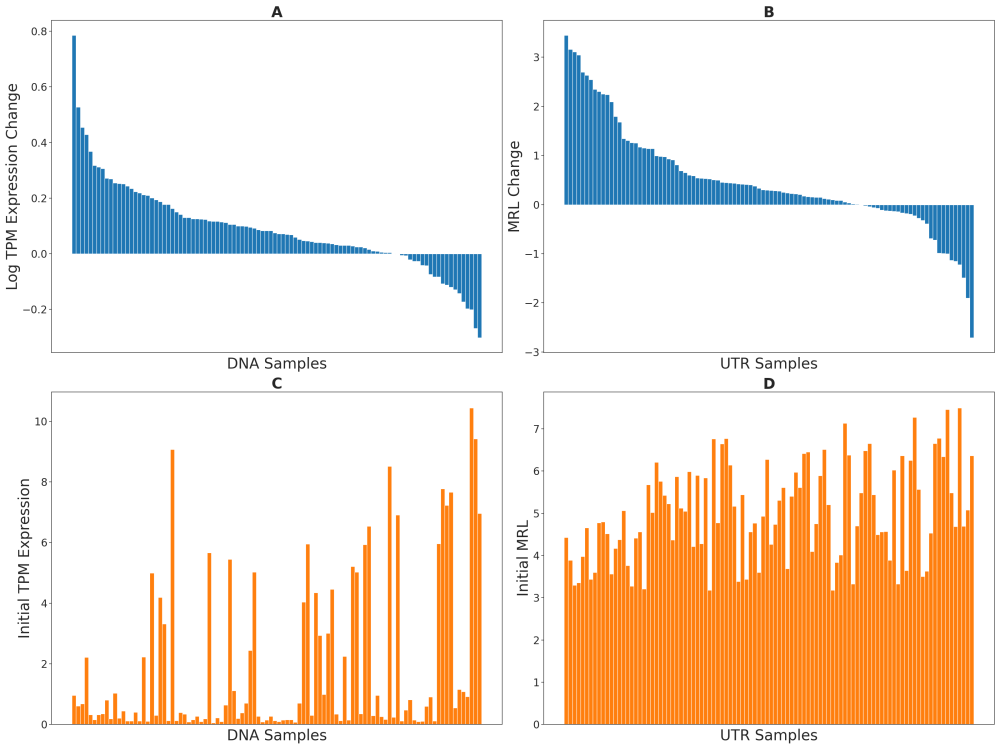

Figure 4: **General MRL and gene expression optimization A)** The expression change after 3000 iterations of optimization for 100 DNA samples shows that the model successfully generates 5' UTR sequences and optimizes a majority of those to improve the original expression. The values are log TPM Expression values, and the optimization yields higher gene expression in 80% of the DNAs. **B)** Here is shown the mean ribosome load change of 100 sequences is optimized. In this case, a 70% majority of the sequences yield substantially higher MRL after optimization. **C and D)** Here, we show the initial MRL and gene expression values for the corresponding samples in panels A and B, respectively. For both panels, initial values tend to increase towards the right. We see that it is more likely for the optimization to degrade performance for samples with high initial MRL or gene expression values.

3 Discussion

We introduce UTRGAN, the first framework for developing synthetic 5' UTR sequences optimized towards a target metric. Our approach enables us to generate sequences with similar attributes to the human 5' UTR samples without the need for exploring intractable search space of DNA sequences that have lengths of up to 128 nucleotides. We advance the state-of-the-art, which relies on optimizing existing UTRs via modifications and by generating sequences from scratch. Our model can also generate sequences of variable lengths, further increasing the possible space of generated sequences.

UTRGAN is a robust framework for synthetic 5' UTR engineering and is applicable to numerous applications for genetic circuit design. The model's components consist of a GAN adopted to be trained on the human natural 5' UTR dataset and state-of-the-art prediction models for MRL and mRNA abundance [20][19]. Our model can potentially optimize sequences for any objective function as long as a differentiable prediction model exists to predict the target metric for a 5' UTR sequence. Similarly, the model can work with classification models to generate sequences of

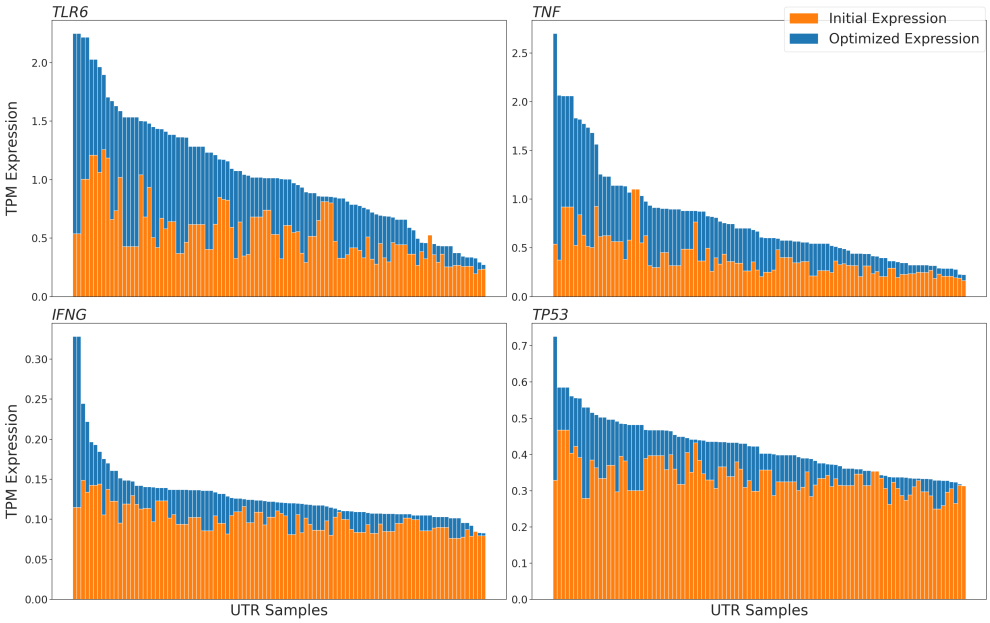

Figure 5: In addition to optimizing for random genes, it is possible to optimize many generated UTR sequences for specific genes. Here is shown the result of optimization for four specific genes. TLR6, IFNG, TNF, and TP53 genes show a substantial increase in the expression for optimized sequences compared to initial predicted expression values. The optimization shows 90%, 100%, 35%, and 29% increase in expression on average for *TLR6*, *TNF*, *IFNG*, and *TP53*, respectively.

a particular class if the prediction model is replaced with such a classifier.

Although UTRGAN can generate and optimize sequences with desirable characteristics, the optimization can lead to decreased MRL or mRNA in less than 30% of the cases for MRL and 20% of the cases for mRNA abundance. One possible reason for decreased expression is the limitation with regard to the predictor model (Xpresso). It operates on the entire DNA (gene) sequence, and while back-propagating the gradient to update the sequence, we can only use part of the gradient that corresponds to the 5' UTR. Thus, changing the 5' UTR but not the rest of the sequence might lead the model to end up at a worse point in the loss space. For both the expression and the MRL optimization, we observe that the decreased scores tend to occur when the optimization procedure starts with a sequence that yields a high score, as seen in Figure 4 (Panels C and D). In such a case, users can use the starting sequence and discard the optimization.

We expect to see wide use of data-driven sequences generated using deep learning methods in the near future. Generating 5' UTRs utilizing this approach is practical and does not require massive computational resources. While we focus on 5' UTRs, it is straightforward to generalize the framework to generate 3' UTRs or other regulatory elements. It is also possible to use other generative models, such as variational autoencoders or diffusion models. Generating sequences with regulatory roles enables researchers to conduct experiments in a much shorter time, which can be particularly beneficial in mRNA vaccine production and mRNA-based therapeutics.

4 Methods

4.1 Datasets and the Experimental Setup

We obtain the UTR dataset from the UTRdb [30], a complete dataset of 3' and 5' UTR sequences of various organisms. We use only the human 5' UTR samples. The length of the sequences ranges from 30 bps to 600 bps. We limit the training data to sequences of length between 64 bps and 128 bps. To train and validate the UTRGAN model, we use 8,280 and 900 natural 5' UTR samples, respectively, and use a batch size of 64 sequences.

We obtain the DNA sequences of the genes used for optimization from the Ensembl Biomart tool [31]. The gene expression prediction model used here requires sequences of fixed 10,500 lengths, and we collect the data accordingly. These sequences correspond to all genes in the human genome. The Xpresso model [19] only 3,500 bps downstream of the TSS and 7,000 upstream of the TSS. To ensure the resulting sequence is not shorter than 10,500 nts after replacing the natural 5' UTR, we get 128 more nucleotides downstream of the TSS.

We encode the data using one-hot encoding, and the dimension of each UTR is $(128, 5)$, where 128 is the maximum size, as mentioned, and 5 is the one-hot vector. The DNA alphabet in one-hot encoded as follows: A:(1,0,0,0,0), C:(0,1,0,0,0), G:(0,0,1,0,0), T:(0,0,0,1,0), X:(0,0,0,0,1) where X is the padding character used when the sequence is shorter than 128 bps. This padding is necessary because the input of the discriminator (critic) as a convolutional neural network has to be in fixed dimensions, and the sequences naturally vary in length. In addition, to generate truly random sequences, we select a random length between 64 and 128 nucleotides and then select each nucleotide randomly from the DNA alphabet.

The model is implemented using the Tensorflow 2 [32] library and in Python. We train and optimize the model on a Super-Micro Super-Server 4029GP-TRT with 2 Intel Xeon Gold 6140 Processors (2.3 GHz, 24.75 M cache), 256 GB RAM, 6 NVIDIA GeForce RTX 2080 Ti GPUs (11 GB, 352 bit), and 2 NVIDIA TITAN RTX GPUs (24 GB, 384 bit). We only use a single TITAN RTX GPU. The training takes around 120 hours for 200,000 epochs, and the MRL and gene expression optimization take 4 and 10 minutes, respectively. The code is available at https://github.com/ciceklab/UTRGAN.

4.2 The Model

Overall Architecture

The generative model used here is a variant of a Generative Adversarial Network (GAN) [33]. The input of the GAN is a random noise sample z, and we train the GAN to generate realistic 5' UTR sequences. The GAN learns to map the noise to the 5' UTR space. Then, we can optimize the model to generate specific sequences by optimizing the input noise only. The optimization performed here does not update the weights of the generator and changes the input noise z instead.

The sequences generated by the GAN are scored by the selected scoring models. We use two deep convolutional neural networks to score the generated sequences. The mRNA abundance scoring model [19] predicts the log TPM expression of a given gene sequence, including a 5' UTR. The model used to predict the ribosome load of the 5' UTR requires only the 5' UTR sequence [20] as input. Both models and the generator are differentiable, which enables us to optimize the input z with Stochastic Gradient Ascent by updating it in multiple iterations.

4.2.1 Generative Model Architecture

The GAN used here is based on the original Wasserstein GAN (WGAN) [18] model that is an improved version of the Deep Convolutional GAN (DCGAN) model [17] with the Wasserstein loss [34]. However, there are challenges to training a Wasserstein GAN in a stable manner. To ensure stable training of the network, we use an improved training procedure of the WGAN using Gradient-Penalty [16].

The layers of the deep learning model are modified to fit this task based on the dimensions of the data, with 128 nucleotides, each represented by a vector of length 5. The generator consists of a dense layer to increase the dimension of the input noise, following five residual convolution blocks, all using the same number of channels. Finally, the output of the last residual block is fed to a convolution layer with five output channels following a Softmax layer, the output of which is the one-hot encoded sequence. See Supplementary Table 2 for layer dimension details. The input is a random vector of length 40, sampled from a normal distribution with a mean of zero and a standard deviation of 1.0. Technically called critic, the discriminator of this model is a convolutional neural network with five residual blocks following a dense layer with one output (See supplementary table 3 for details). Unlike the original DCGAN model [17], there is no activation function at the output of the dense layer of the critic, as the output is an unbounded score and not a probability [18]. The number of iterations of the critic per generator iteration is 5 here, meaning that the critic is updated 5 times more than the generator.

The model consists of a generator G and a critic D. The generative gets a random noise as input and generates random samples, and the weights of the network are initialized with a normal distribution range of -0.1 and 0.1. The model's loss function, as shown in Equation 1, is to maximize the distance between the critic's scores for the actual sequences and the generated ones. The first part of the loss function belongs to the original WGAN [18], and the second part includes the gradient penalty and its coefficient λ. Here x and $\tilde{x}$ are the actual and generated samples, where we sample $\hat{x} \sim P_{\tilde{x}}$ uniformly along straight lines between pairs of points sampled from the data distribution $\mathbb{P}r$ and the generator distribution $\mathbb{P}g$ for the computation of the gradient penalty. The gradient penalty improves the training of the GAN by enforcing the Lipschitz constraint as the model proposes [16], and this allows training for a higher number of iterations. Here we train our model for 200,000 epochs.

$$L_{adv} = (\mathbb{E}_{\tilde{x}\sim P_g}[D_{\tilde{x}}] - \mathbb{E}_{x\sim P_r}[D_x]) + \lambda(\mathbb{E}_{\hat{x}\sim P_{\hat{x}}}\left[(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2\right]) \tag{1}$$

4.2.2 Expression Prediction Optimization

As mentioned above, the Xpresso model can predict the expression of a one-hot encoded DNA sequence with a fixed length of 10,500 nucleotides. The model receives sequences of 3,500 nucleotides downstream and 7,000 nucleotides upstream of a Transcription Start Site (TSS) and predicts the log TPM expression for the sequences.

Let z_0 denote the initial latent vector, then the generator inputs the latent vector ($G(z_0)$) and generates the 5' UTR, which we call U_g. S denotes the gene sequence for which we design the 5' UTR with the original 5' UTR U_r. The expression prediction model is denoted as $E(S)$ where S is the input sequence.

To carry out the optimization, the initial 5' UTRs are first generated using a fixed seed for the random latent vector. They are attached to the randomly selected gene sequences replacing the original 5' UTRs, and the initial expression is measured. We define R as the function to replace the original 5' UTR U_r with the generated 5' UTR U_G while maintaining the length and the TSS

position on the sequence S (see Equation 2). The original position of the TSS in the DNA samples is the 7000th nucleotide from the start. The length of a sequence is obtained via the *len* function. We use the bracket notation to slice the sequences.

$$R(U_g, S) = S[0:7000] + U_g + S[7000 + len(U_g) : (10500 - (len(U_g) - len(U_r)))] \tag{2}$$

The mean saliency scores in the Xpresso paper [20] show the significance of the effect of nucleotides around the TSS coordinate on the predicted expression. The plot shows the highest effect to be on the downstream of the TSS where the 5' UTR is placed. Based on this analysis of the Xpresso model, we expect the optimization of the 5' UTR part of the DNA alone to increase the expression value.

To perform the Gradient Ascent to the input latent vector for optimization, we first calculate the gradient of the output of the Xpresso model with respect to its input DNA sequences (Equation 3) and slice the part of the gradient that corresponds to the 5' UTR from the 7000th to the 7128th index where the 5' UTR is located. In the next step, the gradient of the generator is calculated with respect to the input noise vector. Updating the noise vector according to the gradient is supposed to increase the expression in the following iteration. This iteration is repeated up to 3,000 times by updating the input latent vector (see Equation 4) until the optimization converges, meaning that it does not increase after 100 iterations.

$$\frac{\partial E(R(G(z_0), S))}{\partial z_0} = \frac{\partial E(R(G(z_0), S))}{\partial R(G(z_0), S)}[7000:7128] \times \frac{\partial G(z_0)}{\partial z_0} \tag{3}$$

$$z_{new} = z_0 + \frac{\partial E(R(G(z_0), S))}{\partial z_0} \tag{4}$$

4.2.3 Mean Ribosome Load Prediction Optimization

The FramePool model was trained on sequences of length up to 100 bps, and utilizing the Frame-slice layer allows it to predict the MRL value for 5' UTR sequences of any length [20]. Frame-slicing, which slices the sequence into 3-mers or codons referred to as frames and applies global pooling on the indices, making it size-independent. To perform the optimization with this model, again, we use the Gradient Ascent algorithm. Here, slicing is not required, and the derivatives are taken directly on the 5' UTR sequence. Considering the MRL prediction model as M and the initial latent vector as z_0 again, the gradient is calculated as shown in Equation 5 and applied as shown in Equation 6. This is repeated for 1,000 iterations to obtain the optimized 5' UTRs.

$$\frac{\partial M(G(z_0))}{\partial z_0} = \frac{\partial M(G(z_0))}{\partial G(z_0)} \times \frac{\partial G(z_0)}{\partial z_0} \tag{5}$$

$$z_{new} = z_0 + \frac{\partial M(G(z_0))}{\partial z_0} \tag{6}$$

References

[1] Ulmer, J. B. & Geall, A. J. Recent innovations in mrna vaccines. *Current opinion in immunology* **41**, 18–22 (2016).

[2] Andronescu, M., Fejes, A. P., Hutter, F., Hoos, H. H. & Condon, A. A new algorithm for rna secondary structure design. *Journal of molecular biology* **336**, 607–624 (2004).

[3] Busch, A. & Backofen, R. Info-rna—a server for fast inverse rna folding satisfying sequence constraints. *Nucleic acids research* **35**, W310–W313 (2007).

[4] Sav, S., Hampson, D. J. & Tsang, H. H. Simard: A simulated annealing based rna design algorithm with quality pre-selection strategies. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 1–8 (IEEE, 2016).

[5] Liang, Y., Huang, L. & Liu, T. Development and delivery systems of mrna vaccines. *Frontiers in Bioengineering and Biotechnology* **9** (2021).

[6] Lu, T. K.-T., Kellis, M., Cao, J., Pardo, E. M. N. & Zhang, Z. Synthetic 5'utr sequences, and high-throughput engineering and screening thereof (2020). US Patent App. 16/441,647.

[7] Von Niessen, A. O. *et al.* 3-utr sequences for stabilization of rna (2022). US Patent 11,492,628.

[8] Von Niessen, A. O. *et al.* 3'-utr sequences for stabilization of rna (2019). US Patent App. 15/763,709.

[9] Burkhardt, D. H., Subramanian, R. R. & Cobaugh, C. Heterologous utr sequences for enhanced mrna expression (2022). US Patent 11,389,546.

[10] Killoran, N., Lee, L. J., Delong, A., Duvenaud, D. & Frey, B. J. Generating and designing dna with deep generative models. *arXiv preprint arXiv:1712.06148* (2017).

[11] Zrimec, J., Buric, F., Kokina, M., Garcia, V. & Zelezniak, A. Learning the regulatory code of gene expression. *Frontiers in Molecular Biosciences* **8** (2021).

[12] Zrimec, J. *et al.* Supervised generative design of regulatory dna for gene expression control. *BioRxiv* (2021).

[13] Zrimec, J. *et al.* Gene expression is encoded in all parts of a co-evolving interacting gene regulatory structure. *bioRxiv* 792531 (2019).

[14] Trotta, E. On the normalization of the minimum free energy of rnas by sequence length. *PloS one* **9**, e113380 (2014).

[15] Beck, J. D. *et al.* mrna therapeutics in cancer immunotherapy. *Molecular Cancer* **20**, 1–24 (2021).

[16] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. & Courville, A. C. Improved training of wasserstein gans. *Advances in neural information processing systems* **30** (2017).

[17] Radford, A., Metz, L. & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[18] Arjovsky, M., Chintala, S. & Bottou, L. Wasserstein generative adversarial networks. In *International conference on machine learning*, 214–223 (PMLR, 2017).

[19] Agarwal, V. & Shendure, J. Predicting mrna abundance directly from genomic sequence using deep convolutional neural networks. *Cell reports* **31**, 107663 (2020).

[20] Karollus, A., Avsec, Ž. & Gagneur, J. Predicting mean ribosome load for 5'utr of any length using deep learning. *PLoS computational biology* **17**, e1008982 (2021).

[21] Karras, T., Laine, S. & Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4401–4410 (2019).

[22] Vinh, L. V., Lang, T. V., Binh, L. T. & Hoai, T. V. A two-phase binning algorithm using l-mer frequency on groups of non-overlapping reads. *Algorithms for Molecular Biology* **10**, 1–12 (2015).

[23] Konu, Ö. & Li, M. D. Correlations between mrna expression levels and gc contents of coding and untranslated regions of genes in rodents. *Journal of Molecular Evolution* **54**, 35–41 (2002).

[24] Noreen, M. & Arshad, M. Association of tlr1, tlr2, tlr4, tlr6, and tirap polymorphisms with disease susceptibility. *Immunologic Research* **62**, 234–252 (2015).

[25] Sun, Y. *et al.* Interferon-γ is protective in cytokine release syndrome-associated extrapulmonary acute lung injury. *American Journal of Respiratory Cell and Molecular Biology* (2022).

[26] Benson, L. N., Liu, Y., Deck, K., Mora, C. & Mu, S. Ifn-γ contributes to the immune mechanisms of hypertension. *Kidney360* **3**, 2164–2173 (2022).

[27] Ma, K., Zhang, H. & Baloch, Z. Pathogenetic and therapeutic applications of tumor necrosis factor-α (tnf-α) in major depressive disorder: a systematic review. *International journal of molecular sciences* **17**, 733 (2016).

[28] Lee, Y.-J. *et al.* Gene expression profiling of glioblastoma cell lines depending on tp53 status after tumor-treating fields (ttfields) treatment. *Scientific reports* **10**, 1–14 (2020).

[29] Ozaki, T. & Nakagawara, A. Role of p53 in cell death and human cancers. *Cancers* **3**, 994–1013 (2011).

[30] Grillo, G. *et al.* Utrdb and utrsite (release 2010): a collection of sequences and regulatory motifs of the untranslated regions of eukaryotic mrnas. *Nucleic acids research* **38**, D75–D80 (2010).

[31] Cunningham, F. *et al.* Ensembl 2022. *Nucleic Acids Research* **50**, D988–D995 (2021). URL `https://doi.org/10.1093/nar/gkab1049`. `https://academic.oup.com/nar/article-pdf/50/D1/D988/42058123/gkab1049.pdf`.

[32] Abadi, M. *et al.* {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 265–283 (2016).

[33] Goodfellow, I. *et al.* Generative adversarial networks. *Communications of the ACM* **63**, 139–144 (2020).

[34] Frogner, C., Zhang, C., Mobahi, H., Araya, M. & Poggio, T. A. Learning with a wasserstein loss. *Advances in neural information processing systems* **28** (2015).