Understanding Transcriptional Regulatory Logic Using Convolutional and Generative Deep Learning Models

Nic Fishman[1,#], Georgi K. Marinov[2], and Anshul Kundaje[2,3]

[1] *Department of Bioengineering, Stanford University, Stanford, California 94305, USA*
[2] *Department of Genetics, Stanford University, Stanford, California 94305, USA*
[3] *Department of Computer Science, Stanford University, Stanford, California 94305, USA*
[#] *To whom correspondence should be addressed: njwfish@stanford.edu*

Abstract

The concerted action of sequence-specific transcription factors onto *cis*-regulatory elements in DNA is the core mechanism through which gene regulation is accomplished in most eukaryotes. Mechanistic dissection of the logic of these interactions is therefore of critical importance for understanding cellular responses to external and internal stimuli in a wide variety of contexts, such as organismal development and disease. However, despite many decades of intensive studies the goal of mapping out THIS *cis*-regulatory logic is still far from achieved. Massively Parallel Reporter Assays (MPRA) are a potentially extremely powerful tool that can be applied for this purpose, but extracting *cis*-regulatory interactions from their output is not trivial for the human eye. Here we apply deep learning approaches to MPRA datasets to understand the individual contribution to gene expression activity of transcription factor (TF) motifs and the relationships between them, and show that convolutional neural networks (CNNs) are capable of learning both aspects of *cis*-regulatory logic. We then build on our CNN models and develop generative adversarial networks (GANs) that can produce novel regulatory sequences with particular gene expression activities.

1. Introduction

Gene regulation in most eukaryotes is driven by the action of sequence-specific transcription factors, which recognize specific DNA sequence motifs/binding sites (TFBSs) within larger *cis*-regulatory elements (cREs), and act in concert to affect the expression of their cognate genes. These cREs typically contain multiple binding sites for several different transcription factors; it is the integration of the activity of all these factors within an individual cRE that determines its activity, and the integration of the activity of all cREs that together act on a gene that drives changes in its expression. The complexity of *cis*-regulatory logic is considerable – each human gene is associated with on average about a dozen putative cREs[1], each of them several hundred bases in length and potentially containing dozens of TFBSs – thus it is not surprising that despite decades of research in the field, understanding that logic in detail has remained an elusive target.

Two technological and analytical developments hold great promise for resolving that challenge.

The first one is the development of MPRAs, which allow the regulatory activity of many thousands of both endogenous as well as synthetic DNA sequences to be assayed simultaneously on a large scale[2–5]. In a typical MPRA design, each tested sequence is either associated with a barcode placed within a reporter gene[2,3,5], or is directly located in the reporter itself[4]. High-throughput DNA sequencing is used to determine the relative abundance of barcodes or input sequences in both the populations of expressed RNA and input DNA molecules, thus providing information on the relative activity of each construct.

The second advance involves the application of deep learning techniques to discerning gene regulatory logic, with the hope that machine learning will prove to be more effective where humans have not succeeded so far. Deep learning approaches have already demonstrated remarkable performance on a wide variety of problems in genomics[6], such as predicting transcription factor binding, accessible chromatin regions, nucleosome positioning, RNA splicing outcomes, and many others, and they have more recently also seen initial applications to MPRAs involving endogenous[8] and random sequences[7]. Once successfully predictive deep learning models are built, methods of deep learning model interpretation can then be used to extract regulatory relationships encoded in the DNA sequence[9].

Beyond the simple development of predictive models, another highly promising deep learning-based approach in this area is the generation of novel DNA sequences (following the "What I cannot create, I do not understand" principle), the activity of which can subsequently be tested in an MPRA. Initial work in this area has shown the ability of generative deep learning methods to generate DNA sequence *de novo*[10], but the success of such approaches in specific biological contexts has not yet been demonstrated.

In this work we first apply deep learning approaches (Figure 1) to an yeast MPRA dataset that includes DNA se-

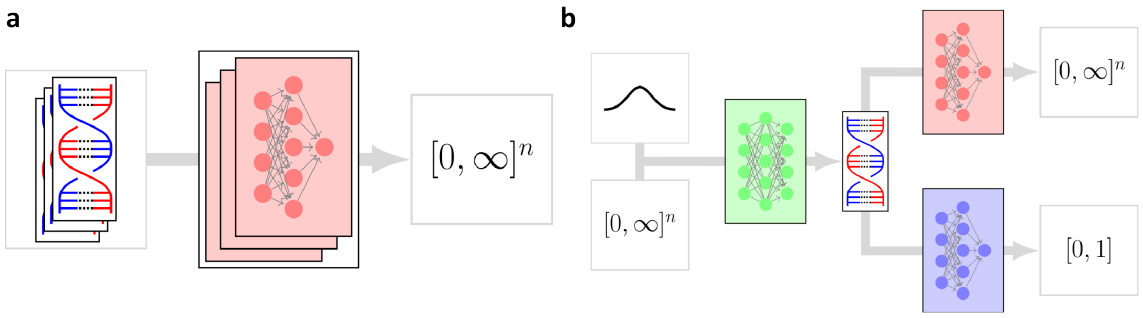

Figure 1: Outline of deep learning strategies for predicting activity and *de novo* generating functionally active DNA sequences. (a) The regression training phase of the sequence generation pipeline takes a set of sequences and trains a number of models to predict expression driven by these sequences (b) Structure of the GAN component of the sequence generation pipeline, which creates *de novo* predicted-to-be functionally active sequences.

quences containing predefined regulatory grammars[12] and their activities at various amino acid concentrations ("[AA]"). We find that CNN-based approaches can successfully capture these lexical grammars, identify relevant motifs, and predict regulatory activities in the context of this MPRA dataset. We build on our CNN models to develop a GAN-based algorithm for generating novel DNA sequences with desired regulatory activities. We then apply these approaches in the context of a much larger MPRA dataset in human cells, and plan on experimentally testing the activity of newly designed regulatory sequences in an MPRA.

2. Related Work

Initial applications to MPRAs involving endogenous[8] and random sequences[7] have shown that deep learning frameworks are capable of capturing the underlying regulatory grammars encoded in DNA sequences. More broadly, approaches applying deep learning to DNA sequence are at this point well developed in the literature.[8,11,17]

The problem of *de novo* sequence generation has been less extensively explored. Killoran et al.[10] applied GANs to generating generic DNA sequences using the Wasserstein GAN, which optimizes the earth mover distance between the generated and real samples[14]. In this approach, the generator was first pre-trained to produce DNA sequences, and then the discriminator was replaced with a differentiable analyzer, oriented towards classifying if a given sequence had a specified property. Along a similar vein, Gupta and Zou[19] develop a GAN approach based on a feedback loop, generating DNA sequence in a generator-discriminator framework, but adding the selection of some "top" sequences (evaluated according to a given criterion) to the "real" distribution each iteration. In this way they expand the "real" distribution and force the generator to construct sequences in line with the analyzer's selection criteria. More recently Brookes et. al. propose a framework for design by adaptive sampling[20], which allows updating a generative model to maximize the output of an oracle, which can be used for sequence design problems of this type.

Here we present a novel framework for end-to-end generation of sequences to minimize any developer-defined loss function on a set of DNA sequences with associated scores for some property(s) of interest.

3. Methods

The backbone of this project is an end-to-end sequence generation pipeline that takes a set of sequences, and some corresponding property(s) to learn, then trains a GAN that can generate sequence designed to exhibit the given property(s). Throughout this paper we examine this pipeline in the context of predicting the regulatory effects of DNA sequences on gene expression, guided by existing MPRA datasets.

Table 1: Architectural search of deep learning CNN models for predicting regulatory activity from sequence. Each architectural property and the corresponding distribution it was drawn from in the random search are shown. Note that for properties relevant to multiple layers, such as units per dense layer, n independent samples were drawn from the distribution corresponding to the relevant number of layers.

Architecture Property	Distribution Drawn From
Number Convolutional Layers	$\sim Uni(2, 4)$
Filters per Convolutional Layer	$\sim Uni(5, 50)$
Filter Size	$\sim Uni(4, 15)$
Number Dense Layers	$\sim Uni(1, 5)$
Units per Dense Layer	$\sim Uni(5, 100)$
Regularize all Layers	$\sim Bern(p = 0.5)$

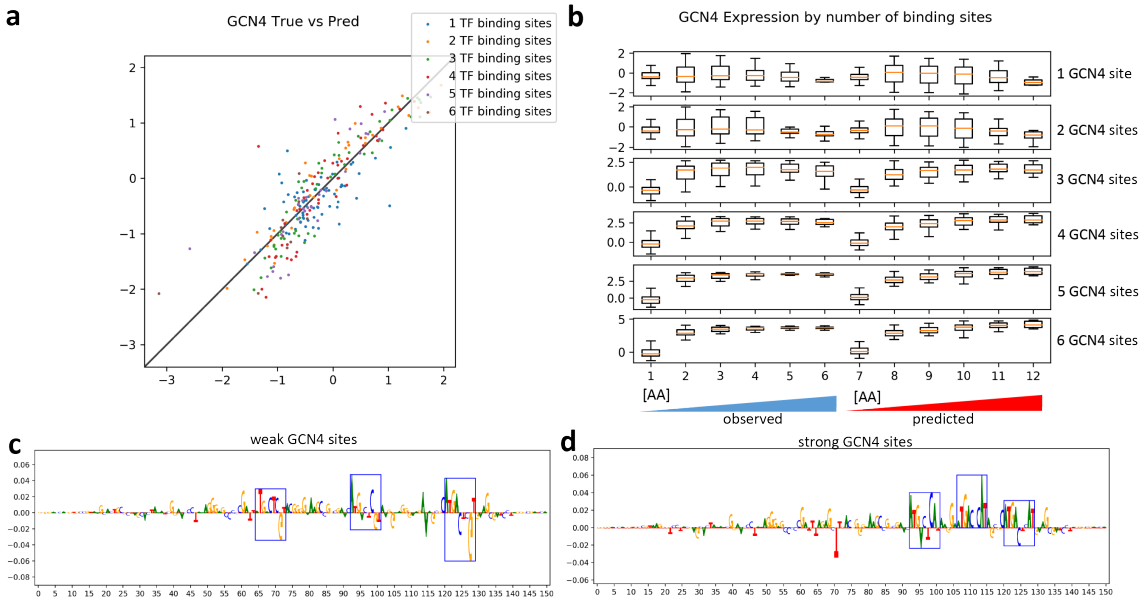

Figure 2: Performance of predictive models on yeast MPRA dataset. (a) True (x-axis) and predicted (y-axis) expression levels for constructs with different numbers of GNC4 motifs. (b) Deep learning models reproduce experimentally the observed behaviors of constructs with different numbers of GNC4 motifs in response to increasing [AA]. (c-d) DeepLIFT importance score for a construct containing three weak GCN4 TFBSs (c) and another one containing three strong GCN4 TFBSs correctly identify motifs driving regulatory activity

3.1. MPRA Datasets and data preprocessing

We used two MPRA datasets in this study. The first one[12] was carried out in the budding yeast *Saccharomyces cerevisiae* and includes testing the activity of ∼5,000 synthetic sequences containing defined numbers of TFBSs for factors known to be important in regulating gene expression upon changes in nutrient availability. Regulatory activity was measured under a range of six different increasing amino acid concentrations.

The Sharpr-MPRA dataset[13] is significantly larger, containing ∼500,000 endogenous 145 bp-long human sequences that tile ∼15,000 putative regulatory regions at 5-bp intervals. These sequences were assayed in two different human cell lines (the erythroid K562 and the hepatic HepG2) with two different promoters.

DNA sequences were one-hot encoded following established practices. We also $log+$-transformed the regression targets so we could use the ReLU non-linearity as the output of our regression networks.

3.2. Predicting Gene Expression

The first part of the pipeline involves building a neural net that learns to predict the effects on gene expression of a given set of regulatory sequences. We largely follow CNN-based approaches previously developed to tackle such tasks[8,11]. After one-hot encoding, the input sequences are split into training and test sets. Convolutional layers learn to capture transcription factor binding motifs, followed by a series of fully connected layers to predict output expression under a set of different conditions; in this case the regressor predicts a vector where each element corresponds to one of the conditions.

Because it is not *a priori* clear exactly which architecture is best suited for a given prediction task, we implement a random search, training 100 models with random numbers of convolutional and dense layers, randomly sized kernels and random numbers of hidden units (Table 1). We train these models for 1,000 epochs using the Adam optimizer, keeping only the lowest validation loss across the training history. Models are then evaluated on the test set, and a linear combination of the test error and the overfitting ratio (defined as $\frac{MSE_{train}}{MSE_{test}}$) is used to rank the regression models. Later in the process it becomes important that the regressor is actually learning to assign importance to TFBSs, and that overfitting is minimized as much as possible.

3.3. Feature Importance Scoring

The DeepLIFT framework[9] was used to assign feature importance scores of MPRA prediction models to each nucleotide in input sequences.

3.4. Generating Regulatory Sequence

To *de novo* generate regulatory sequences, we start with the improved Wasserstein GAN[14,15], the core architecture of which we modify for our purposes in two ways. First, in addition to the random seed z, we also input a continuous vector t, with the same shape as the output from the regressor trained in the previous phase. The vector t is drawn from a normal distribution fit to the distribution of expressions in the MPRA dataset being trained on. The generator outputs a one-hot encoded sequence vector, making use of the

Table 2: Top models from random architecture search for the Yeast MPRA and Sharpr-MPRA datasets. For feature importance scoring, we sum the DeepLIFT importance scores ascribed to TFBSs, and divide by the total importance of the input sequence. We sum this score over all sequences in both the training and test datasets.

Yeast MPRA					Sharpr-MPRA		
Model Rank	Training MSE	Test MSE	Motif Importance		Model Rank	Training MSE	Test MSE
1	0.014043	0.026653	1087.418475		1	8674.099	34120.833
2	0.002005	0.028467	1084.981529		2	7581.866	34191.550
3	0.008275	0.030267	11z77.912115		3	12927.603	34713.798

gumbel-softmax activation function to remain differentiable. The discriminator is trained in classic WGAN-GP fashion, feeding examples from both real sequences from the MPRA dataset and generated ones from the generator. Formally this loss is calculated as:

$$L_D = \mathop{\mathbb{E}}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathop{\mathbb{E}}_{x\sim\mathbb{P}_r}[D(x)] + \lambda \mathop{\mathbb{E}}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}[(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2] \tag{1}$$

Where we update the discriminator by maximizing with respect to L_D. For the generator, we feed the generated sequence into both the discriminator and the regressor (from the previous phase), and the generator is updated by minimizing with respect to a weighted average of the two losses. These losses are weighed with a tunable γ term as follows:

$$\mathcal{L}_\mathcal{G} = \gamma L_D + (1-\gamma) L_R \tag{2}$$

3.5. Evaluating Generated Sequences

3.5.1. Motif Counts

The simplest way to evaluate generated sequences is to check whether they contain TFBSs known to be relevant for gene regulation under the conditions tested in the training set. To do this we can simply count frequency of each motif or motif combination in generated sequences, and compare it to either the real data or randomly generated sequence.

3.5.2. 1-Nearest Neighbor

The best method for evaluating the quality of a generated distribution given an actual distribution is to do calculate the LOO (Leave-One-Out)accuracy of a 1-NN algorithm in a learned feature space.[16] We create a general framework for generating a learned feature space to compare regulatory sequences of a given length $k <= 2000$ by starting with a CNN model,[17] altering the length of the input space, and then truncating the model after the last convolutional layer that produces a non-negative output size. To compute the LOO accuracy, we label real sequences 1 and generated sequences -1, calculate the distances between all sequence pairs, and take the label of the second smallest distance sequence as the predicted label for all sequences. If the generated and real distributions were the same, i.e. if the generated sequences are high quality, the 1-NN will achieve 50% accuracy.

3.5.3. Predicting Expression

To ensure that the generated sequences are not simply overfitting the regressor network used to train the generator, we use an ensemble of several of the next-best regressors produced by the random architecture search to predict the expression of the generated sequences, and compare it to the initial target expression t.

4. Results

4.1. Hyperparameters

Throughout training we used the default hyperparameters for the Adam optimizer, the default hyperparameters for the WGAN-GP algorithm,[15] and a standard batch size of 64, because these parameters are relatively well supported in the literature and returned reasonably good results in preliminary tests.

4.2. Regression

4.2.1. Yeast MPRA

After random architecture search on the yeast MPRA dataset, we arrived at an optimal model with a test MSE of 0.0266 (Table 2). The performance of this model can be compared to the baseline established in the yeast MPRA study itself by physical thermodynamic models for predicting expression, the top test performance of which is $R^2 = 0.8$ across all sequences. The corresponding test R^2 for the top deep learning method is $R^2 = 0.91$ (Figure 2a). Also, a key result of the original study is that combining large numbers of TFBSs in the same construct does not necessarily lead to higher regulatory activity, and can even decrease it, due to competition for binding between nearby sites. Deep learning models are well capable of capturing that behavior (Figure 2b).

To confirm that the neural net's models are not just learning to predict expression, but are also identifying the key regulatory sequences driving, we applied DeepLIFT featue importance scoring[9] on the input sequences. The relevant motif indeed tend to be the most highly scoring sequences, with two examples shown in Figure 2c-d. To more generally quantify motif importance, we also calculate a motif importance score, calculated as the sum of the importance ascribed to basepairs within motif divided by the total importance for

	Metric	Optimal/Real Value	Generated
$\gamma = 1.0$	1-NN LOO	0.5	0.65
	Motif Count Per Sequence	17.54	14.114
	Predicted Expression MSE	0.0	0.801
	Metric	Optimal/Real Value	Generated
$\gamma = 0.5$	1-NN LOO	0.5	0.77
	Motif Count Per Sequence	17.54	11.894
	Predicted Expression MSE	0.0	0.205
	Metric	Optimal/Real Value	Generated
$\gamma = 0.0$	1-NN LOO	0.5	1.0
	Motif Count Per Sequence	17.54	8.336
	Predicted Expression MSE	0.0	0.792

Table 3: Evaluation of sequences produced from 5000 epochs of training for various γ on the Yeast MPRA dataset.

a given sequence. This score is summed over all sequences in both the training and test datasets.

4.2.2. Sharpr-MPRA

The Sharpr-MPRA dataset is significantly more difficult to predict than the Yeast MPRAs as it exhibits poor reproducibility between the experimental replicates[8], and the between-replicate correlation sets an upper limit on the achievable model performance. We are still in the process of identifying the optimal performing models for this dataset.

4.3. Sequence Generation

4.3.1. Yeast MPRA

We were able to successfully train several GANs using the Yeast MPRA regressor, but are still working evaluating and understanding the resulting sequences. Current summary results are shown in Table 3, where several interesting observations are evident. First, using just the WGAN architecture ($\gamma = 1$), the distribution of generated sequences is very similar to the real one, and the per-sequence motif occurrence is close to that in the real data; however, the MSE is large. In contrast, using just the regressor ($\gamma = 1$) results in sequence with poor motif occurrence characteristics. It is only for the combination loss ($\gamma = 0.5$) that we see generated sequences that are approaching the desired properties. This supports the usefulness of the architecture we propose in Fig 1.

An issue we have encountered in this context is that extremely extended training periods (10,000-20,000 epochs) begin to seriously degrade performance on all metrics. We are still investigating its sources and possible solutions.

4.4. Sharpr-MPRA

We only have very preliminary sequence generation results on the Sharpr-MPRA dataset (Table 4), as we have not yet arrived at an optimal prediction model.

5. Conclusions

Here we show, using the Yeast MPRA dataset, that first, our CNN regression architecture is capable of learning regulatory activity from MPRA datasets, and even surpasses the baseline established by van Dijk using physical models, and second, that our proposed modified WGAN deep learning architecture is capable of *de novo* generating DNA sequence using the measured regulatory activities of known sequences as input. As the Sharpr-MPRA dataset is much larger and more difficult to predict, we are still in the process of finding the optimal approach for predicting and generating sequences on it, and this is where the next steps for this project are primarily oriented towards. Training WGAN with various γ values, and annealing the γ parameter over time are among the strategies we are focusing on next.

Additionally, the suite of evaluation tools we have built can be applied to any generated DNA sequences, and comparing against existing sequence generation approaches[10,19,20] is also something we plan to do in the immediate future. We are also interested in establish a more straightforward, non-GAN baseline by setting up an MLE method for sequence generation based on the promise such approaches have shown in the language generation literature.

6. Contributions

N.F., A.K., and G.K.M. conceived the project. N.F. carried out deep learning model generation and data analysis with

Metric	Optimal/Real Value	Generated
1-NN LOO	0.5	0.89
Predicted Expression MSE	0.0	64324.5176

Table 4: Evaluation of sequences produced from 5000 epochs of training for $\gamma = 0.5$ on the SHRAPR MPRA dataset.

input from G.K.M. and A.K. N.F. prepared the manuscript with input and oversight from G.K.M. and A.K.

References

1. ENCODE Project Consortium. 2012. An integrated encyclopedia of DNA elements in the human genome. *Nature* **489**(7414):57–74.
2. Patwardhan RP, Hiatt JB, Witten DM, Kim MJ, Smith RP, May D, Lee C, Andrie JM, Lee SI, Cooper GM, Ahituv N, Pennacchio LA, Shendure J. 2012. Massively parallel functional dissection of mammalian enhancers in vivo. *Nat Biotechnol* **30**(3):265–270.
3. Melnikov A, Murugan A, Zhang X, Tesileanu T, Wang L, Rogov P, Feizi S, Gnirke A, Callan CG Jr, Kinney JB, Kellis M, Lander ES, Mikkelsen TS. 2012. Systematic dissection and optimization of inducible enhancers in human cells using a massively parallel reporter assay. *Nat Biotechnol* **30**(3):271–277.
4. Arnold CD, Gerlach D, Stelzer C, Boryń ŁM, Rath M, Stark A. 2013. Genome-wide quantitative enhancer activity maps identified by STARR-seq. *Science* **339**(6123):1074–1077.
5. Sharon E, Kalma Y, Sharp A, Raveh-Sadka T, Levo M, Zeevi D, Keren L, Yakhini Z, Weinberger A, Segal E. 2012. Inferring gene regulatory logic from high-throughput measurements of thousands of systematically designed promoters. *Nat Biotechnol* **30**(6):521–530.
6. Ching T, Himmelstein DS, Beaulieu-Jones BK, Kalinin AA, Do BT, Way GP, Ferrero E, Agapow PM, Zietz M, Hoffman MM, Xie W, Rosen GL, Lengerich BJ, Israeli J, Lanchantin J, Woloszynek S, Carpenter AE, Shrikumar A, Xu J, Cofer EM, Lavender CA, Turaga SC, Alexandari AM, Lu Z, Harris DJ, DeCaprio D, Qi Y, Kundaje A, Peng Y, Wiley LK, Segler MHS, Boca SM, Swamidass SJ, Huang A, Gitter A, Greene CS. 2018. Opportunities and obstacles for deep learning in biology and medicine. *J R Soc Interface* **15**(141). pii: 20170387.
7. Cuperus JT, Groves B, Kuchina A, Rosenberg AB, Jojic N, Fields S, Seelig G. 2017. Deep learning of the regulatory grammar of yeast 5′ untranslated regions from 500,000 random sequences. *Genome Res* **27**(12):2015–2024.
8. Movva R, Greenside P, Shrikumar A, Kundaje A. 2018. Deciphering regulatory DNA sequences and noncoding genetic variants using neural network models of massively parallel reporter assays. *bioRxiv*:393926.
9. Shrikumar A, Greenside P, Kundaje A. 2017. Learning Important Features Through Propagating Activation Differences. *arXiv*:1704.02685
10. Killoran N, Lee LJ, Delong A, Duvenaud D, Frey BJ. 2017. Generating and designing DNA with deep generative models. *arXiv*:1712.06148 [cs.LG]
11. Zeng H, Edwards MD, Liu G, Gifford DK. 2016. Convolutional neural network architectures for predicting DNA-protein binding. *Bioinformatics* **32**(12):i121–i127.
12. van Dijk D, Sharon E, Lotan-Pompan M, Weinberger A, Segal E, Carey LB. 2017. Large-scale mapping of gene regulatory logic reveals context-dependent repression by transcriptional activators. *Genome Res* **27**(1):87–94.
13. Ernst J, Melnikov A, Zhang X, Wang L, Rogov P, Mikkelsen TS, Kellis M. 2016. Genome-scale high-resolution mapping of activating and repressive nucleotides in regulatory regions. *Nat Biotechnol* **34**(11):1180–1190.
14. Arjovsky, Martin and Chintala, Soumith and Bottou, Léon. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875*
15. Gulrajani, Ishaan and Ahmed, Faruk and Arjovsky, Martin and Dumoulin, Vincent and Courville, Aaron C. 2017. Improved training of wasserstein gans. *Advances in Neural Information Processing Systems* 5767–5777.
16. Qiantong Xu and Gao Huang and Yang Yuan and Chuan Guo and Yu Sun and Felix Wu and Kilian Weinberger. 2018. An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv*:1806.07755.
17. Zhou, J. and Troyanskaya, O.G., 2015. Predicting effects of noncoding variants with deep learning-based sequence model. *Nature methods*, 12(10), p.931.
18. Killoran, N., Lee, L.J., Delong, A., Duvenaud, D. and Frey, B.J., 2017. Generating and designing DNA with deep generative models. *arXiv preprint* arXiv:1712.06148.
19. Gupta, A. and Zou, J., 2018. Feedback GAN (FBGAN) for DNA: a Novel Feedback-Loop Architecture for Optimizing Protein Functions. *arXiv preprint* arXiv:1804.01694.
20. Brookes, D.H. and Listgarten, J., 2018. Design by adaptive sampling. arXiv preprint arXiv:1810.03714.
21. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016, November. Tensorflow: a system for large-scale machine learning. In OSDI (Vol. 16, pp. 265-283).
22. Chollet, F., 2017. Keras (2015).
23. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), pp.2825-2830.
24. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L. and Lerer, A., 2017. Automatic differentiation in pytorch.