Targeted optimization of regulatory DNA sequences with neural editing architectures

Anvita Gupta[1], **Anshul Kundaje**[1,2]
[1] Department of Computer Science [2] Department of Genetics
Stanford University
{avgupta, akundaje} @ stanford.edu

Abstract

Targeted optimizing of existing DNA sequences for useful properties, has the potential to enable several synthetic biology applications from modifying DNA to treat genetic disorders to designing regulatory elements to fine tune context-specific gene expression. Current approaches for targeted genome editing are largely based on prior biological knowledge or ad-hoc rules. Few if any machine learning approaches exist for targeted optimization of regulatory DNA sequences.

Here, we propose a novel generative neural network architecture for targeted DNA sequence editing - the EDA architecture - consisting of an encoder, decoder, and analyzer. We showcase the use of EDA to optimize regulatory DNA sequences to bind to the transcription factor SPI1. Compared to other state-of-the-art approaches such as a textual variational autoencoder and rule-based editing, EDA significantly improves predicted binding of SPI1 of genomic sequences with the minimal set of edits. We also use EDA to design regulatory elements with optimized grammars of CREB1 binding sites that can tune reporter expression levels as measured by massively parallel reporter assays (MPRA). We analyze the properties of the binding sites in the edited sequences and find patterns that are consistent with previously reported grammatical rules which tie gene expression to CRE binding site density, spacing and affinity.

1 Introduction

Recent generative models for genomic sequences, such as generative adversarial networks, variational autoencoders, and recurrent neural networks, have largely focused on ab initio generation of biological sequences from distributions learned over a large collection of exemplar sequences[10, 6, 7]. However, generative models have been shown to suffer from low diversity - falling into the failure mode of producing generic sequences with high likelihood [8]. Generative models that are capable of editing an existing sequence, rather than generating an entirely new sequence from scratch, may be able to draw from the natural diversity present in biological sequences, while still allowing useful changes to the data. Also, many applications of genome editing typically require editing an existing DNA sequence in order to knock out or repair disease genes or modify regulatory DNA to modulate gene expression in specific cell types and states.

Machine learning approaches for editing existing sequences for desired properties have been significantly less well studied than generative models. Guu *et al* proposed a neural editor for natural language to transform an input sentence into an output based on a sampled edit vector; however, the edit vectors are latent and must be interpreted after training [8].

Here, we propose a novel Encoder-Decoder-Analyzer (EDA) neural network architecture that radically departs from status quo methods [8] by combining recurrent sequence-to-sequence models, latent vectors based on an explicit predictor, and adversarial example generation techniques to automatically generate candidate modifications to regulatory DNA sequences to optimize specific properties such as binding of transcription factors or reporter gene expression. This represents a unique possibility to leverage existing supervised learning models that map regulatory sequences to specific properties, to instead design sequences via editing with desired properties. We showcase the EDA model on two pilot case studies. In the first case study, we use EDA to edit regulatory DNA sequences to increase the binding probability of a transcription factor SPI1 by leveraging *in vivo* genome-wide binding profiles (ChIP-seq) for SPI1. In the second case study, we use EDA to generate candidate regulatory sequences containing binding sites of the CREB1 transcription factor that can produce a desired gene expression readout as measured by a massively parallel reporter assay (MRPA) from Davis *et al*, 2019 [3]. The EDA approach significantly outperforms existing state-of-the-art approaches.

2 Methods

Sequence Variational Autoencoder (SVAE) as a baseline method: The sequence variational autoencoder (SVAE) for editing is based off the recurrent architecture described in [1]; the encoder produces the parameters (μ, Σ) of a Gaussian distribution in latent space, from which z, a latent vector encoding the sequence x, is sampled. The decoder attempts to reconstruct the input sequence from z. The loss function of the VAE is given in Equation 1. For editing, the latent space of the SVAE was perturbed through the addition of Gaussian Noise ($\hat{z} = z + e$) where $e \in \mathcal{N}(0, 0.4)$ as proposed in Guu *et al* [8].

$$\mathcal{L}_{\text{VAE}}(\theta, x) = -\mathbb{E}_{q_\theta(z|x)}(\log(p_\phi(x|z))) + \text{KL}(q_\theta(z|x)|p(z)) \tag{1}$$

Encoder Decoder Analyzer (EDA) Custom Architecture Our novel architecture called EDA consists of three deep neural network components: an Encoder, Decoder, and Analyzer. The Encoder and Decoder are recurrent neural networks (RNN) with attention. Conceptually the Encoder can transform any one-hot encoded input DNA sequence to a compact latent representation. The Decoder can generate an output DNA sequence given a specific instantiation of the latent representation learned by the encoder. The Analyzer is a neural network that can map the latent representation of a DNA sequence to a specific property that we are typically interested in optimizing. Here, we use convolutional neural networks (CNN) as the Analyzer architecture, although it can be any differentiable architecture. The procedure for editing in the EDA architecture consists of three phases.

Stage 1: Training Encoder-Decoder. The Encoder-Decoder seq2seq architecture is trained to reproduce its inputs. The Encoder takes in sequence x and produces a latent space embedding z, while the Decoder takes z and attempts to reproduce the original sequence x.

Stage 2: Training Analyzer The same Encoder from stage 1 is also followed by the Analyzer, which takes in the latent state z of a sequence from the Encoder, and produces an output prediction $\hat{y}$ such as the probability that the given sequence binds to a transcription factor such as SPI1.

Stage 3: Editing. Given input sentence x, the encoder produces the latent state embedding z for the sentence. We update the latent state to minimize the loss function $\mathcal{L}$ (the binary cross entropy loss) between the analyzer's prediction $\hat{y}$ and the desired score y via the Fast Gradient Sign Method (FGSM) [5] as in Equation 2.

$$z' := z - \epsilon \times \text{sign}(\nabla_z \mathcal{L}(\hat{y}, y)) \tag{2}$$

The latent state is updated until the loss is approximately 0 ($L(\hat{y}, \tilde{y}) \sim 0$). The decoder produces the edited sequence $\hat{x}$ from the modified latent representation z'. Epsilon (ϵ) is a hyperparameter varying between zero and one corresponding to the size of steps taken in the latent space.

The pseudocode for the EDA training is shown in Algorithm 1. The training algorithm includes an additional step where the binding score is predicted from the edited sentence $\hat{x}$. This step is necessary as the latent representation $\tilde{z}$ of the decoded sequence $\hat{x}$ may not be the same as the perturbed latent representation z' due to noise in the decoding process.

Algorithm 1 EDA Architecture Editing.

```
Require: Encoder $E$, Analyzer $A$, and Decoder $D$ are pre-trained; $y$ is the desired label, $x$ is given sequence.
    $z := E(x)$; $\hat{y} := A(z)$; Loss := $\mathcal{L}(A(z), y)$
    $z' := z$
    while Loss > tolerance do
        $z' := z' - \epsilon \times sign(\nabla_z \mathcal{L})$
        Loss := $\mathcal{L}(A(z'), y)$
    end while
    Edited sentence $\hat{x} := D(z')$
    Re-encode edited sentence $\tilde{z} := E(\hat{x})$
    Final prediction $\tilde{y} := A(\tilde{z})$
```

3 Optimizing regulatory DNA sequences for binding of the SPI1 transcription factor

Dataset 43,787 reproducibly-identified peaks from an ENCODE ChIP-seq experiment on lymphoblastoid cell line GM1278 (GEO GSM803531) were used as the positive labeled set of putative SPI1 bound sequences [2]. The negative labeled set was constructed from an equal number of non-overlapping unbound 200bp sequences from the human genome.

Baselines An SVAE model was trained as a neural baseline, as described above. A simple rule-based editing model was also constructed that randomly adds the SPI1 consensus binding site ("AGGAA") if not already present in the sequence.

Evaluation Method Edited sequences are evaluated on three quantitative metrics: similarity to the original DNA sequence, predicted binding score (probability) of SPI1, and percent of sequences with matches to known SPI1 binding motifs [9]. Binding score is measured by an independent CNN model trained to discriminate 200bp sequence labeled as bound by SPI1 ChIP-seq data from a balanced number of background sequences from the geneome. This independent CNN model achieves AUROC of 0.979 and AUPRC of 0.978 on a held-out test set. Similarity of edited sequences to the original sequences was calculated by the gapped kmer-mismatch (GKM) kernel, which evaluates DNA sequence similarity based on gapped kernel overlap [4]. We also used the BLEU-4 score, a metric more commonly used in NLP translation as another measure of sequence similarity.

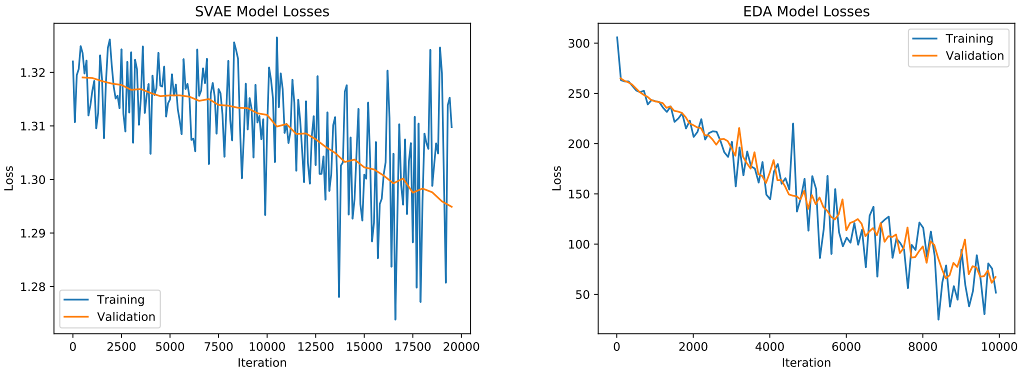


Figure 1: Training and validation loss curves of EDA Architecture (left) and VAE architecture (right).

	GKM Similarity	BLEU-4	% Predicted Positive	% Binding Site
EDA $\epsilon = 0.55$	**0.524**	**86.8**	73.4	31.8
EDA $\epsilon = 0.99$	0.491	85.1	**84.4**	**34.2**
VAE	0.052	33.12	16.0	20.6
Rule Baseline	0.965	98.3	45.0	100

Table 1: Comparison of Editing Methods in terms of sequence similarity (out of 1), BLEU-4 score (out of 100), and percentage of sentences predicted positive.

3.1 SPI1 Editing Results

Training Results The loss curve for the SVAE, as well as the Encoder-Decoder portion of the EDA Architecture is shown in Figure 3.1. The Encoder-Decoder of EDA achieves average edit distance of 8.8 from input to output sequence after 20,000 iterations of training. The accuracy of the analyzer in the EDA architecture is 92.674%, with AUROC of 0.979 and AUPRC of 0.978.

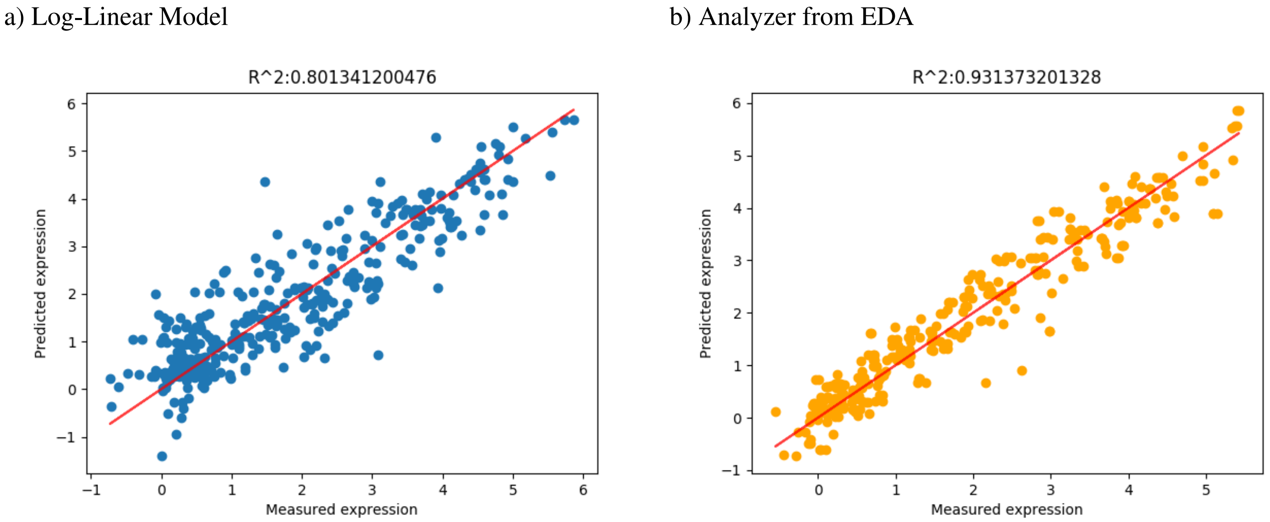
EDA Edited Sequences. 500 randomly selected sequences from the balanced test set were edited, with about half the sequences originally labeled as positive (i.e. bound based on overlap with SPI1 ChIP-seq peaks). Results are shown in **Table 1**. Edits from the EDA architecture demonstrate high similarity (52.39% on average) to the original sequences, whereas the SVAE edits achieve similarity of only 5.2%. Any two random sequences from the set have GKM similarity of 2.048%. Thus, rather than editing, the SVAE appears to be sampling separate sequences.

Overall, 84.4% of EDA-edited sequences are predicted to bind to SPI1 by the independent model. 34.2% of these sequences contain the full SPI1 binding site, and 63.4% of sequences contain the "GGAA" portion, which is most highly conserved in the SPI1 PWM [9]. The rule-based baseline achieves only 45% sequences predicted positive, similar to the chance that any randomly chosen test set sequence would be predicted positive. Thus, editing the sentence to optimize for binding score is more complex than simply inserting known high affinity SPI1 binding sites.

Table 2 shows a DNA sequence which initially had a low binding score on the independent model, whose edit received a high score. The EDA model modifies the area in the initial sequence in gray into the full SPI1 binding motif shown in orange.

Original	**Edited**
Predicted Score: 0.0891	Predicted Score: 0.9947
CCCCAGATTCGGAATTGTGATTTAAACTAG GCCTCTCTTCAGGCATCAGAACCAAATTAG AGTGCTTTATACGTGCAAGGAACTAATGCA GAAAGACATTATGGGTAAGTC AAGGGA GA TGCCCCAGCCATTGGGCAAATTCCTCTGAC TCCTCCCTGATGGAACTAATTCTTTTTCTTC TGTGCCCGGATCCTAGAATC	CCCGACTTCGGCAATTGGAATTGGAATCAG CCGCTCTTCACCTGCTAGACCGACAACATT GTGGATTTCATTGCAGCGAAGCAGTAATCG AGAAGACATGTTAGCTTTAAC GAGGAA CT GGCCGACATCATGGCGACTTCTCTCATCGC GCTCACTGTGAGAAGCTACTTTTTTGTTCTT CTGTGCGCCGCAGAACTTGA

Table 2: Original sequence and edited sequence from the EDA architecture. The SPI1 motif is highlighted in orange.


a) Log-Linear Model

b) Analyzer from EDA

Figure 2: Predicted expression versus measured expression for a) independent log-linear model, and b) analyzer from EDA architecture

4 Optimizing Reporter Expression in CREB1 MPRA Dataset

Dataset The CRE MPRA dataset from Davis *et al* correlates reporter gene expression with CRE binding site strength, number, spacing, and distance from the promoter region. The genomic MPRA dataset consists of 3480 sequences of 150bp within 3 backgrounds with different combinations and locations of CREB1 binding sites. The strong CREB1 consensus binding site is "TGACGTCA", and the weak binding site is "TGAAGTCA", where the method to generate this dataset is fully described in [3]. Genetic expression here is measured by the ratio of RNA barcode reads of a sequence to the count of DNA reads; for predictive purposes, a log transform was performed on the data, and a histogram of log expression levels is shown in figure **??**.

From analysis of the MPRA dataset, Davis *et al* find four main correlations between sequence and expression levels: 1) number of strong CRE binding sites largely determines expression, 2) weak binding sites increase expression given the presence of at least one strong binding site, 3) higher expression occurs with lower distance of CRE binding sites to the promoter region, and 4) spacing between CRE binding sites modulates periodicity of expression, as two strong binding sites are moved along the sequence.

Here, the editing task is to optimize sequences for particular expression profiles; in particular, to edit MPRA sequences that initially have high expression ($log(\text{expression})) \geq 5$) to low expression, and vice versa. As several correlations between sequence and expression are already discussed by Davis *et al*, we may investigate whether the edited sequences show evidence of these rules which would validate that they are likely to possess the desired expression profile after editing.

Independent Analyzer As an independent predictor from the analyzer in the EDA architecture, we train a log-linear model of expression levels similar to Davis *et al*, with expression predicted from the number of strong and weak binding sites, sequence background, average spacing between CREB1 sites, and distance from the minimal promoter element; in addition, polynomial features of degree 2 were used to model interaction terms. This simple model achieves $R^2 = 0.801$ on a held out test set, and was used to evaluate the edited sequences from the EDA model; correlation between predicted and measured expression is shown in Figure 4a.

Training Results The components of the EDA model were trained on the CRE MPRA dataset to learn a latent representation of the sequences through the encoder-decoder portion, and to predict reporter expression levels from this latent representation through the analyzer. As above, the encoder and decoder consisted of a one layer GRU, where the decoder had sofftmax attention over the encoder outputs, and achieved an average edit distance of 6.06 between input and output after training for 10,000 iterations. The analyzer, as above, was a CNN architecture with three convolutional layers, each followed by a ReLU activation, average pooling, and two dense layers; this architecture achieved $R^2 = 0.93$ on a held out test set; the analyzer's predicted expressions correlate well with measured expressions, as shown in Figure 4b.

EDA Editing Results Here, our editing task is to invert the expression profile of CREB1 MPRA sequences from the test set, and to evaluate whether the resulting MPRA sequences displayed known patterns of CRE binding site placement elucidated by Davis *et al*. In particular, 204 total sequences having $log(\text{expression}) <= 0$ are edited to a higher expression level, with target

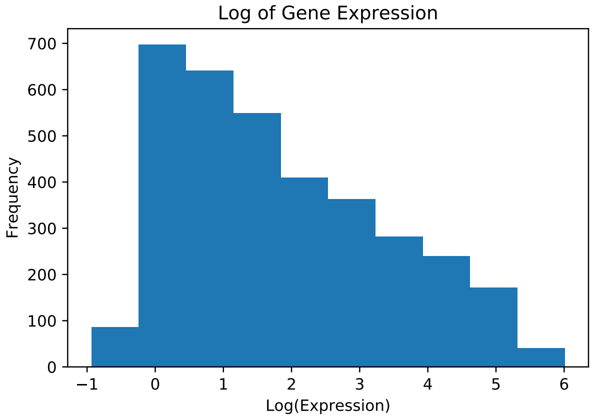


Figure 3: Histogram of log(expression) levels for sequences in CREB1 MPRA dataset; expression ranges widely, from less than zero, meaning that number of RNA barcoded reads are less than the original DNA reads, to six.

set at $log(\text{expression}) = 5.0$, and 96 total sequences with $log(\text{expression}) >= 5$ are edited for a lower level of expression, with target set at $log(\text{expression}) = 0$.

As evaluated by the independent analyzer, 79.4% of sequences to be edited from low expression to high expression are actually predicted to have higher expression than their initial levels. 100% of sequences edited from high to low expression, meanwhile, have lower predicted expression than their initial levels. The histograms of expression before and after editing are shown in Figure 4.

Several representative examples are also evaluated from the edited sequences in order to evaluate presence of correlations described in the original CREB1 MPRA results, and are shown in Figure 4. In particular, example 1 illustrates the result that number of strong CREB1 binding sites are correlated with expression levels, as the edited sequence has the third strong CREB1 binding site changed to a weak site, resulting in expression being predicted to be more than 2x lower after editing.

Also, in example 2, the initial sequence has low expression (-.101) and four weak binding sites; after editing, the third weak binding site is changed to a strong binding site, the fourth weak binding site is deleted, and four additional strong binding sites are added to the sequence. This edit, with predicted expression 4.58, also displays the result found in Davis *et al*, that number of weak binding sites increases reporter gene expression given the presence of at least one strong binding site.

The third example, also going from high expression to low expression, has the initial binding site moved further away from the minimal promoter element, which aligns with reported grammatical rules from the CRE MPRA dataset, in which higher distance from the promoter has been correlated with lower expression. These examples are presented with the caveat that they cannot be used to show that the model has "learned" particular rules - only that the results from the EDA architecture align with known experimental correlations between CRE1 binding sites and reporter expression from the MPRA results.

5 Conclusion

The EDA architecture proposed for targeted genomic editing brings together a broad array of techniques from attention-based seq2seq models, adversarial example generation, and computer vision. The architecture leverages existing genomic predictors to generate candidate modifications of sequences with multiple properties, namely: high probability of SPI1 binding, and desired gene expression profiles for sequences based on CREB1 binding MPRA data. In the SPI1 case, we have explicitly compared the EDA model to existing neural baselines - such as the Sequence VAE model and a rule-based baseline - and shown that the architecture vastly improves upon existing models in both predicted binding affinity and similarity of original to edited sequences.

In the initial CREB1 results, we have shown that on an independent model, high percentages of the edited sequences show the desired shift in expression. Furthermore, several edited sequences display known correlations between CRE binding site strength, number, and position - demonstrating that the edited sequences are consistent with known biological rules about how transcription factor binding relates to reporter expression in this assay. In the future, such architectures could accelerate the pace of research by combining experimental advancements with advancements in AI for targeted genomic editing.

	Initial Sequence	**Edited Sequence**
Example 1	Log(Expression) = 5.2121	Log(Expression) = 2.82
	GCTGACGTCAATGCACACGC ACAGAGCTGACGTCAATTAA ACACGCACATGCTGACGTCA ATTATCAACGGCACCGCTGA CTTCAATCTTTACATGTGGTA GCTAACTAACTATCCTACTGT GCCTGGTCTCACCGACTGCT TTCCCTGG	GCTGACGTCAATGCACACGCA CAGAGCTGACGTCAATTATCA ACGGCACCGCTGACTTCAATC TGTCTGGTGCTTTCCCTGGTGC GCTGTCTGGTGCTTTCCCTGGT GCCTGGTCTCTGACTGGTGCT TTCCCTGGTGCCTGGTCTCTGA
Example 2	Log(Expression) = -0.105	Log(Expression) = 4.585
	GCTGACTTCAATGCACACGC ACAGAGCTGACTTCAATTAA ACACGCACATGCTGACTTCA ATTATCAACGGCACCATCTG CTCTCAGCTTTACATGTGGT GCTGACTTCAATATCCTACT GTGCCTGGTCTCACCGACTG CTTTCCCTGG	GCTGACTTCAATGCACACGCA CAGAGATGCTGACTTCAATTA CACGGCAGCTGACGTCAATCA TTATCAAGGCTGACGTCAATC TGTCTGAACGGCTGACGTCAA TCTGTCTGCTGACGTCATGTGT CTGAACGCTGACGTCAATTGC TT
Example 3	Log(Expression) = 5.2121	Log(Expression) = 2.82
	GCTGACGTCAATGCACACGC ACAGAGCTGACGTCAATTAA ACACGCACATGCTGACTTCA ATTATCAACGGCACCGCTGA CGTCAATCTTTACATGTGGT GCTGACTTCAATATCCTACT GTGCCGCTGACTTCAATCTG CTTTCCCTGG	GCTGACATGGTGTCTGAAAGA ATGGCATGGCTGACGTCAATG TGTCTGGTGCTGTCTGAAAGA ATGGCATGGCTGACGTCAATT GTCTGGTGCTGTCTGAAAGAA TGTCTGGTAGCGCTGACGTCA TGACGTCAATGTCTGGTAGCG CTG

Figure 4: Strong CREB1 binding motif is highlighted in yellow, while weak motif is highlighted in orange. Predicted expression, measured by the log of the ratio of RNA to DNA counts, is shown above each sequence.

References

[1] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

[2] E. P. Consortium et al. An integrated encyclopedia of dna elements in the human genome. *Nature*, 489(7414):57, 2012.

[3] J. E. Davis, K. D. Insigne, E. M. Jones, Q. B. Hastings, and S. Kosuri. Multiplexed dissection of a model human transcription factor binding site architecture. *bioRxiv*, 2019. doi: 10.1101/625434. URL `https://www.biorxiv.org/content/early/2019/05/02/625434`.

[4] M. Ghandi, D. Lee, M. Mohammad-Noori, and M. A. Beer. Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS computational biology*, 10(7):e1003711, 2014.

[5] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *stat*, 1050:20, 2015.

[6] A. Gupta and J. Zou. Feedback gan for dna optimizes protein functions. *Nature Machine Intelligence*, 1(2):105, 2019.

[7] A. Gupta, A. T. Müller, B. J. Huisman, J. A. Fuchs, P. Schneider, and G. Schneider. Generative recurrent networks for de novo drug design. *Molecular informatics*, 37(1-2):1700111, 2018.

[8] K. Guu, T. B. Hashimoto, Y. Oren, and P. Liang. Generating sentences by editing prototypes. *Transactions of the Association of Computational Linguistics*, 6:437–450, 2018.

[9] S. Heinz, C. Benner, N. Spann, E. Bertolino, Y. C. Lin, P. Laslo, J. X. Cheng, C. Murre, H. Singh, and C. K. Glass. Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and b cell identities. *Molecular cell*, 38(4):576–589, 2010.

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