RaptGen: A variational autoencoder with profile hidden Markov model for generative aptamer discovery

Natsuki Iwano [1], Tatsuo Adachi [2], Kazuteru Aoki [2], Yoshikazu Nakamura [2] and Michiaki Hamada [1,3,4*]

1 Graduate School of Advanced Science and Engineering, Waseda University, Tokyo, Japan
2 RIBOMIC, inc., Tokyo, Japan
3 Computational Bio Big-Data Open Innovation Laboratory (CBBD-OIL), National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
4 Graduate School of Medicine, Nippon Medical School, Tokyo, Japan

Abstract

Nucleic acid aptamers are generated by an *in vitro* molecular evolution method known as systematic evolution of ligands by exponential enrichment (SELEX). A variety of candidates is limited by actual sequencing data from an experiment. Here, we developed RaptGen, which is a variational autoencoder for *in silico* aptamer generation. RaptGen exploits a profile hidden Markov model decoder to represent motif sequences effectively. We showed that RaptGen embedded simulation sequence data into low-dimension latent space dependent on motif information. We also performed sequence embedding using two independent SELEX datasets. RaptGen successfully generated aptamers from the latent space even though they were not included in high-throughput sequencing. RaptGen could also generate a truncated aptamer with a short learning model. We demonstrated that RaptGen could be applied to activity-guided aptamer generation according to Bayesian optimization. We concluded that a generative method by RaptGen and latent representation are useful for aptamer discovery. Codes are available at `https://github.com/hmdlab/raptgen`.

1 Introduction

Aptamers are short single-stranded oligonucleotides that bind to specific targets through their three-dimensional folding structure. They are analogous to antibodies and have a variety of applications, including therapeutics [1, 2], biosensors [3], and diagnostics [4]. The advantages of aptamers are that they are rapidly developed by *in vitro* generation, are low immunogenic [5], and have a wide range of binding targets, including metal ions [6], proteins [7], transcription factors [8], viruses [9], organic molecules [10], and bacteria [11]. Aptamers are generated by the systematic evolution of ligands by exponential enrichment (SELEX) [12, 13]. SELEX involves iterations of affinity-based separation and sequence amplification. This iterative process results in an enriched pool that is analyzed for candidate selection. Recent advances in high-throughput sequencing have enabled us to conduct high-throughput SELEX (HT-SELEX) to collect a vast number of aptamer candidates [14–16]. Therefore, computational approaches that efficiently process high-throughput sequencing data are critical in aptamer development.

Several computational approaches have been reported for identifying aptamers using HT-SELEX data. Aptamer identification tools utilize parameters associated with the SELEX principle, such as frequency, enrichment, and secondary structure [17–20]. Although they are useful for identifying sequences from HT-SELEX data, a variation of candidates is limited by the actual sequence existence in the data. Simulation-based methods have been reported for sequence generation [21–23]. These methods require preceding motif information and, therefore, are not suitable for identifying aptamers against an unfamiliar target. Computational approaches have also been developed to predict aptamer motifs. Motif prediction is useful not only for candidate discovery but also for aptamer development processes, such as truncations and chemical modifications. Several methods have been developed for motif detection using secondary structures [24], enrichment of subsequences during SELEX experiments [25], and emphasis on

*To whom correspondence should be addressed. Tel: +81 3 5286 3130; Fax: +81 3 5286 3130; Email: mhamada@waseda.jp

various loop regions [26]. In addition to these approaches, AptaMut utilizes mutational information from SELEX experiments [22]. As nucleotide substitutions can increase aptamer affinity, mutational information is beneficial for candidate discovery. However, while insertions and deletions are also important factors for altering aptamer activity, *in silico* methods dealing with these mutations are poorly developed. Thus, a method that generates sequences from experimental data is needed to expand exploratory space, and including motif information and nucleotide mutations confer an increased opportunity for aptamer discovery.

To develop a procedure for aptamer generation and motif finding, we focused on a neural network. As reported previously, neural networks are suitable for analyzing large datasets and are compatible with high-throughput sequencing data. DeepBind adopts a convolutional neural network (CNN) to distinguish DNA motifs from transcription factors and find sequence motifs by visualizing network parameters [27]. Recurrent neural networks can also be used for sequence discovery [28, 29]. Currently, neural network-driven generative models are being applied in a broad range of research areas. Some examples of neural network-dependent generative models include deep belief networks (DBNs) [30], variational autoencoders (VAE) [31], and generative adversarial networks (GAN) [32]. For a probabilistic generation of nucleic sequences, using long short-term memory (LSTM) was proposed to mimic sequence distribution [33]. GAN-based sequence generation methods have also been proposed [34].

In small molecule discovery, variational autoencoder (VAE)-based compound designs have been reported. VAEs learn a representation of the data by reconstructing the input data from a compressed vector [31]. Kusner *et al.* used grammar-based VAE and SMILES sequences to generate chemical structures for activity optimization [35], and Gómez-Bombarelli *et al.* utilized the representation to design chemical compounds [36]. Unlike other generative models, the VAE exploits the relationship between compressed feature space and inputs in a bi-directional manner; therefore, it is suitable for visualizing similarity-oriented classifications and emphasizing important sequence features. Utilizing the VAE to convert HT-SELEX data into low dimensional space would be useful for candidate discovery; thus, VAE-based aptamer generation systems are worth investigating. To conduct VAE modeling for HT-SELEX data, the latent space should be favorable to aptamer discovery; it captures motif subsequences, robust with substitutions, deletions, and insertions, and can easily monitor effects from the subsequences.

Here, we present RaptGen, a variational autoencoder for aptamer generation. RaptGen uses a profile hidden Markov model decoder to efficiently create latent space in which sequences form clusters based on motif structure. Using the latent representation, we generated aptamers not included in the high-throughput sequencing data. Strategies for sequence truncation and activity-guided aptamer generation are also proposed.

2 Materials and Methods

2.1 Overall Study Parameters

The VAE proposed in this study is a CNN-based encoder with skip connections and a profile HMM decoder with several training methods. Two simulation datasets containing different types of motifs were generated to assess the interpretability of the decoder. Two independent HT-SELEX datasets were subjected to the VAE, and the Gaussian Mixture Model (GMM) was used for multiple candidate selection. Furthermore, Bayesian optimization (BO) was performed based on the activities of tested sequences proposed by GMM, and sequences were truncated by shortening the model length. The process is explained in detail in the following sections. An overview is shown in Figure 1.

2.2 Architecture of the RaptGen Model

2.2.1 Variational Autoencoder (VAE).

VAEs consist of an encoder neural network that transforms input sequence $\mathbf{x}$ into latent distribution $q_\phi(\mathbf{z}|\mathbf{x})$ and a decoder neural network that reconstructs the input data from latent representation $\mathbf{z}$ by learning $p_\theta(\mathbf{x}|\mathbf{z})$. As VAE is a generative model, it can be evaluated by model evidence. However, given a dataset $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^N$, the model

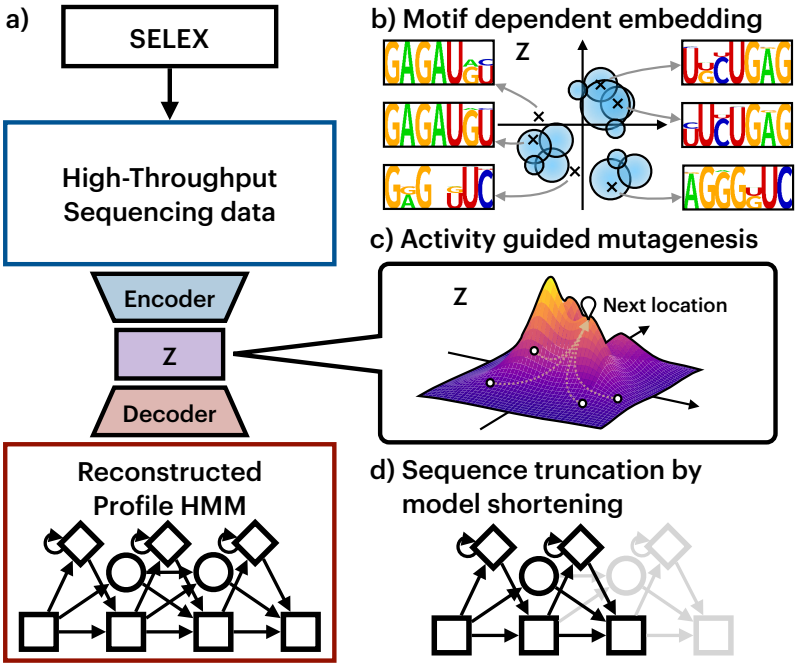


Figure 1. Overall RaptGen schematic and its applications. (a) RaptGen workflow. RaptGen is a VAE with profile HMM for decoder distribution, which considers insertions and deletions. Through training, RaptGen learns the relationship between high-throughput sequencing data and latent space embeddings (the latent space in shown in Z in this figure). (b) RaptGen can construct a latent space based on sequence similarity. It can also generate intermediate representations with no training data. (c) RaptGen can propose candidates according to the activity distribution by transforming a latent representation into a probabilistic model. (d) RaptGen can perform *in silico* sequence truncation by using a short profile HMM decoder.

evidence $p_{\boldsymbol{\theta}}(\mathbf{X})$ is not computationally tractable. Alternatively, we can maximize the Evidence Lower BOund (ELBO); $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{X})$ to calculate how the model describes the dataset using Jensen's inequality,

$$\log p_{\boldsymbol{\theta}}(\mathbf{X}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{X}) = \sum_{i=1}^{N} \mathcal{L}\left(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}\right),$$

where

$$\mathcal{L}\left(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}\right) = -D_{KL}\left(q_{\boldsymbol{\phi}}\left(\mathbf{z} \mid \mathbf{x}^{(i)}\right) \| p_{\boldsymbol{\theta}}(\mathbf{z})\right) + \mathbb{E}_{q_{\boldsymbol{\phi}}\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log p_{\boldsymbol{\theta}}\left(\mathbf{x}^{(i)} \mid \mathbf{z}\right)\right],$$

where $D_{KL}(p||q)$ is the Kullback–Leibler divergence between distributions p and q. The former term on the right-hand-side is the regularization error, and the latter is the reconstruction error. Modeling this reconstruction error to suit the problem determines the structure of the latent space.

2.2.2 CNN-based Encoder With Skip Connections.

The RaptGen encoder network consists of a stack of convolutional layers with skip connections. Each character was first embedded into a 32-channel vector and went through seven convolutional layers with skip connections. Then, max pooling and fully-connected layering transform the vector into the distribution parameters of latent representation $q_{\phi}(\mathbf{z}|\mathbf{x})$. The structure is shown in detail in the Supplementary Information subsection S1.5.

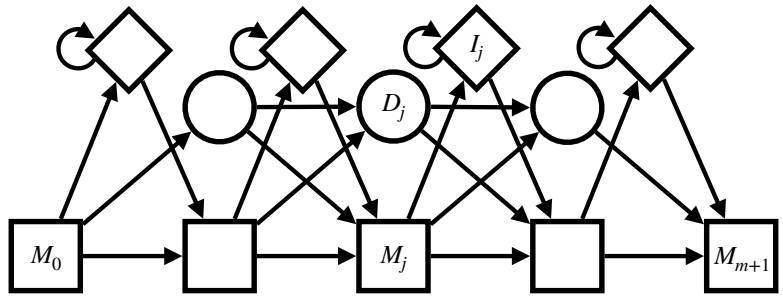


Figure 2. Profile HMM. The squares, circles, and diamonds represent the match M, deletion D, and insertion I states, respectively. The arrows represent possible transition directions of the states. The insertion and deletion states cannot go back and forth between each other. Each matching state emits a character, while the start matching state M_0 and the end matching state M_{m+1} emit a null character.

2.2.3 Profile HMM Decoder Model.

For modeling insertions and deletions, we used the profile hidden Markov Model (profile HMM) as the decoder for RaptGen. The profile HMM is a model that outputs by probabilistically moving from state to state (Figure 2). The profile HMM consists of a match state (M), an insertion state (I), and a deletion state (D). Each state emits specific outputs introduced to represent multiple sequence alignments [37]. The match state has a high probability of emitting a particular character, the insertion state has an equal chance, and the deletion state always emits a null character. These probabilities are called emission probabilities. The other probabilistic parameter is the transition probability. This defines the likeliness of transition from a state to the next state. In a profile HMM, the emission probability $e_S(c)$ is the probability of output character c from state S, and transition probability $a_{S,S'}$ is the probability of changing state from S to S'. These are defined as $e_S(c) = p(c|S)$ and $a_{S,S'} = p(S'|S)$, respectively.

Because profile HMM is a model in which the state transition depends only on the previous single state, the sequence probability $p(\mathbf{x})$ can be written by using the Markov chain rule:

$$p(\mathbf{x}) = \sum_{\pi} p(\mathbf{x}, \pi) = p(x_{0:L+1}, \pi_{\text{last}} = M_{m+1}), \tag{1}$$

where π is the possible state path, π_{last} is the last state in the path, L is the length of the sequence, $x_{j:k}$ is the subsequence of $\mathbf{x}$ from the j-th character to the k-th character on both ends, x_0 is a null character that indicates the start of the sequence, x_{L+1} is a null character that indicates the end of the sequence, and m is the number of matching states in the model. It is computationally expensive to calculate the sequence probability for all possible paths. Introducing a forward algorithm can lower the computational cost to $\mathcal{O}(Lm)$. The forward algorithm consists of a forward variable defined as $f_j^S(i) = p(x_{0:i}, \pi_{\text{last}} = S_j)$, and the probability can be calculated recurrently by

$$\begin{aligned}
f_k^M(l) &= e_{M_k}(x_l) \sum_{S \in \{M,I,D\}} a_{S_{k-1},M_k} f_{k-1}^S(l-1), \\
f_k^I(l) &= e_I(x_l) \sum_{S \in \{M,I\}} a_{S_{k-1},M_k} f_k^S(l-1), \\
f_k^D(l) &= \sum_{S \in \{M,D\}} a_{S_{k-1},M_k} f_{k-1}^S(l).
\end{aligned}$$

The emission probability of the insertion state does not depend on the position of the motif; therefore, it is set to a constant of 1/4 for RNA sequences. We set the probability to output the final end-of-sequence token $p(x_{L+1}|M_{m+1})$ to 1.

2.2.4 Other Tested Decoders.

Three probabilistic models were tested: the multi-categorical (MC) model, auto-regressive (AR) model, and profile HMM. The probabilistic models each have different sequence probability assignments. The MC model assigns a

categorical distribution to each position of the sequence. Given the representation vector $\mathbf{z}$ and the probability of the sequence $\mathbf{x}$, $p(\mathbf{x}|\mathbf{z})$ is calculated by $p(\mathbf{x}|\mathbf{z}) = \prod_{i=1}^{L} p(x_i|z) = \prod_{i=1}^{L} \mathrm{Cat}(x_i|f_\theta(\mathbf{z}))$, where Cat is a categorical distribution and f_θ is a neural network. The AR model outputs a probability according to previous data. The probability of the sequence $p(\mathbf{x}|\mathbf{z})$ is calculated by $p(\mathbf{x}|\mathbf{z}) = \prod_{i=1}^{L} p(x_i|x_{0:i-1}, \mathbf{z}) = \prod_{i=1}^{L} \mathrm{Cat}(x_i|g_\theta(x_{0:i-1}, \mathbf{z}))$, where g_θ is a recurrent neural network (RNN). The architecture of networks f_θ and g_θ is described in the Supplementary Information subsection S1.5.

2.3 Training Techniques.

To learn RaptGen, state transition regularization was introduced. Also, weighed regularization loss was introduced for all VAEs, including RaptGen.

2.3.1 State Transition Regularization.

A VAE can be trained with backpropagation by treating ELBO as a loss function. In addition to ELBO, a Dirichlet prior distribution was used on the transition probabilities to avoid unnecessary state transitions in the early rounds of training RaptGen. By penalizing transitions other than match-to-match at the beginning of the learning process, insertions and deletions are forced to occur less. This allows continuous motifs to be learned and lowers the probability of obtaining models with meaningless transitions traversing deletion states.

The probability of categorical variable $\boldsymbol{p} = \{p_k\}$ sampled from a Dirichlet distribution is

$$\mathrm{Dir}\left(\boldsymbol{p} \mid \boldsymbol{\alpha}\right) = \frac{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)}{\prod_{k=1}^{K} \Gamma\left(\alpha_k\right)} \prod_{k=1}^{K} p_k^{a_k - 1},$$

where $\boldsymbol{\alpha} = \{\alpha_k\}$ is the Dirichlet distribution parameter. The regularization term is the sum of the log-odds ratio of the training probability from the matching state over each position i, defined as

$$\begin{aligned} L_M(\boldsymbol{p}_i, e, r) &= \log\left(\frac{\mathrm{Dir}(\boldsymbol{p}_i \mid \boldsymbol{\alpha}(w_m))}{\mathrm{Dir}(\boldsymbol{p}_i \mid \boldsymbol{\alpha}(0))}\right) \\ &= \log\left(\frac{\Gamma(3+w_m)}{\Gamma(1+w_m)}(a_{M_{i-1},M_i})^{w_m} \times \frac{1}{\Gamma(3)}\right) \\ &= \log\left(\frac{(2+w_m)(1+w_m)}{2}(a_{M_{i-1},M_i})^{w_m}\right), \end{aligned}$$

where $\boldsymbol{p}_i$ is $\begin{bmatrix} a_{M_{i-1},M_i} & a_{M_{i-1},I_i} & a_{M_{i-1},D_i} \end{bmatrix}$ which indicates the transition probabilities from the i-th matching state, and $\boldsymbol{\alpha}(w_m) = \begin{bmatrix} 1+w_m & 1 & 1 \end{bmatrix}$ is the parameter representing the induction weight w_m. To make this loss zero at a specific round R, w_m was set to $4(1 - e/R)$, where e is the training epoch. This regularization term was added to the ELBO during training.

2.3.2 Weighted Regularization Loss.

The scaling parameter for the regularization was introduced to train the VAE. Scaling the regularization term of the loss function of the VAE to minimize the value in the early epoch of training improves latent embedding [38]. The scale is defined as e/E, where e is the training epoch, and E is the maximum number of epochs to have scaling. After the E epochs of training have finished, the scale is set to 1.

2.3.3 Training Settings.

All sequences in the training set were filtered first. Sequences with exact matching adapters, exact matching sequence design lengths, and sequences read more than once remained. The sequences were split into training and test datasets in a 9 to 1 ratio. The model with the smallest test loss was selected through iterations. For the weighted regularization loss, the maximum number to have scaling E was set to 50. The state transition regularization parameter R was set to 50 for the profile HMM decoder. Adam was used as the training optimizer

with default parameters [39]. All the networks were trained up to 2000 epochs with early stopping when the test loss was not updated for 50 epochs.

2.4 RaptGen Evaluation.

2.4.1 Simulation Data.

For the simulation data shown in Figure 3(a), ten different motif sequences of length ten were generated, and single nucleotide modification with a 10% error rate was added. In other words, each motif sequence had a 10/3 percent chance of deletion, insertion, or modification at a specific position. After this procedure, sequences were randomly extended to reach 20 nt by adding nucleotides to the right and the left. We made 10,000 sequences in total, with no duplication.

For the simulation data shown in Figure 4(a), sequences containing paired motifs were generated. Two five-nt motifs were made, and then one of the motifs was randomly deleted at a probability of 25 percent each. If both motifs remained, two to six nt were randomly inserted between the left and right motifs. Subsequently, sequences were randomly extended to reach 20 nt, and 5,000 of these sequences were generated.

2.4.2 SELEX Data.

SELEX data used in this study were obtained previously [20]. The sequences are available as DRA009383 and DRA009384, which we call dataset A and dataset B, respectively. Dataset A consists of nine SELEX rounds from 0 to 8, and dataset B consists of four rounds from 3 to 6. The round with the smallest unique ratio $U(T)$ with the restriction of $U(T) > 0.5$ was used, defined as

$$U(T) = \frac{|\{\mathbf{x}|\mathbf{x} \in \mathcal{D}(T)\}|}{|\mathcal{D}(T)|},$$

where $\mathcal{D}(T)$ are the whole sequences, read in round T. The 4th round was selected for each dataset.

2.5 RaptGen Applications in Aptamer Discovery

2.5.1 GMM for Initial Sequence Selection.

We used the GMM for initial sequence selection from the obtained latent space. To efficiently select ten points to be evaluated, GMM was run 100 times with ten components, and the mean vectors of the model with the best evidence (likelihood) were selected.

2.5.2 Surface Plasmon Resonance (SPR) Assay.

The SPR assays were performed using a Biacore T200 instrument (GE Healthcare) as described previously with slight modifications [20]. The target proteins of datasets A and B were human recombinant transglutaminase 2 (R&D systems, Cat. No. 4376-TG) and human recombinant integrin alpha V beta 3 (R&D systems, Cat. No. 3050-AV), respectively. Aptamers were prepared with fixed primer regions and 16-mer poly(A)-tails as follows: 5' - GGGAGCAGGAGAGAGGUCAGAUG - (variable sequence) - CCUAUGCGUGCUAGUGUGA - (polyA) - 3' for dataset A and 5' - GGGAGAACUUCGACCAGAAG - (variable sequence) - UAUGUGCGCAUACAUGGAUCCUC - (polyA) - 3' for dataset B. Previously reported aptamers were used as positive controls. All evaluated sequences are listed in the Supplementary Information Table S1. Aptamers were prepared by *in vitro* transcription using a mutant T7 RNA polymerase and 2'-fluoro-pyrimidine NTPs. The running buffer consisted of 145 mM NaCl, 5.4 mM KCl, 0.8 mM $MgCl_2$, 1.8 mM $CaCl_2$, 0.05% Tween20, and 20 mM Tris–HCl (pH 7.6). A 5'-biotinylated dT16 oligomer was immobilized to both active and reference flow cells of the streptavidin sensor chip (BR100531, GE Healthcare). The poly(A)-tailed RNA was captured in the active flow cell by complementary hybridization at a concentration of 300 nM and a flow rate of 20 µLmin^{-1}, with an association time of 60 s. The proteins were injected into the flow cells of the sensor chip at a concentration of 50 nM and a flow rate of 20 µLmin^{-1}, with an association time of 60 s. To regenerate the sensor chip, bound aptamers were completely removed by injecting 6 M urea. Data were obtained by subtracting the reference flow cell data from the active flow cell data. The ratio of the protein-binding level to

aptamer-capturing level was used as binding activity. Percent relative binding activities of positive control aptamers are shown in the results and discussion section. For normalization of dataset A, the cycle number-dependent reduction of control aptamer binding was estimated.

2.5.3 Multipoint BO via Local Penalization.

BO uses both the search for sequences that have not been explored to a reasonable extent and the utility of utilizing sequences with known affinity to select the next sequence for evaluation. The local penalization function is a method that can determine the multi-point expected improvement of candidates by considering the smoothness of the potential function [40]. As it converges faster than qEI [41] and other methods for simultaneous optimization. We used this method to perform multi-point optimization. Implementation was performed with the GPyOpt package [42].

3 Results and Discussion

3.1 Motif-Dependent Embeddings Using Simulation Data

3.1.1 Embedding Continuous Motif with Different Decoder Models.

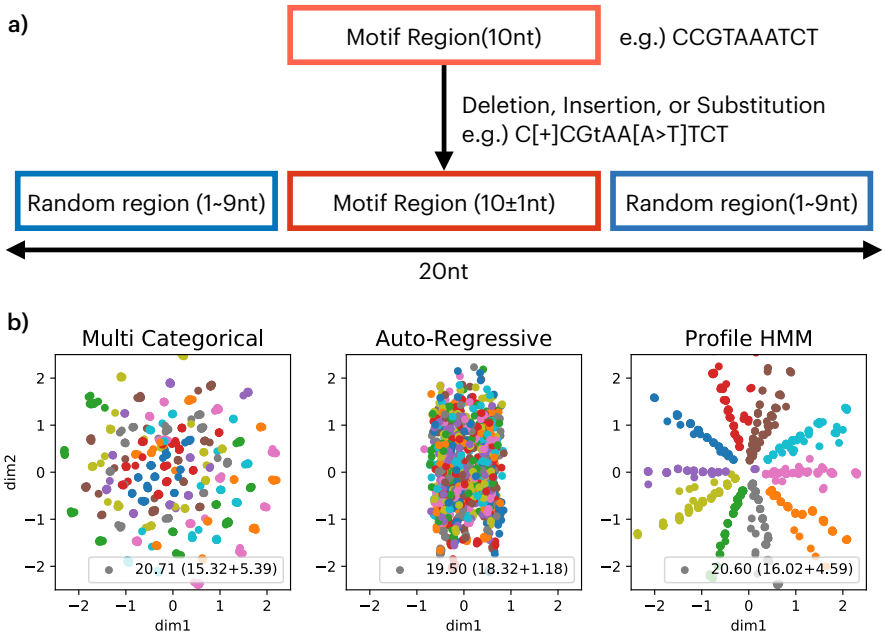


Figure 3. (a) Scheme of simulation data used for evaluating the decoder models. Ten different motifs with a 10% chance of having nucleotide mutations were randomly extended to 20 bases. (b) Embeddings of hypothetical motifs using different decoder models. The simulation data obtained in (a) were subjected to the VAE with the MC, AR, and profile HMM. The resulting latent space is shown. The ELBO is in the right bottom corner with the reconstructed error and KL divergence. Each motif is plotted with different colors.

We first attempted to construct a VAE with an encoder and a decoder applicable to aptamer discovery. In the aptamer representation space, sequences containing the same motif should be in a neighboring area. Robustness against nucleotide mutations and motif positions should also be considered. To identify a desirable decoder, we investigated different types of sequence representation models. We constructed VAEs with a CNN encoder and three different types of probabilistic models: the MC model, AR model, and profile HMM, as a decoder. Simulation data including ten different motifs were created to assess the visualizing capability of these VAEs (Figure 3a). We observed that profile HMM embedded sequences in a motif-dependent manner after training the data, whereas the

MC and AR models displayed indistinctive distributions (Figure 3b). To evaluate model evidence, the ELBO was calculated. Although the MC model and profile HMM had almost the same ELBO (20.71 and 20.60) and had similar reconstitution errors (15.32 and 16.02) and KL divergence scores (5.39 and 4.59), the embedding space of the MC model failed to visualize a motif cluster. This is thought to be due to the inability of the MC model to consider motif positions. Because the nucleotide probability of each position was independently estimated in the MC model, the same motifs in the shifted position might not be aligned in latent space. The AR model had the lowest ELBO (19.50); however, the reconstitution error was the worst (18.32). Furthermore, the classification result was not optimal. We suppose that latent representation is dispensable in the AR model because the model itself has context information. Additionally, we compared the different encoder types. LSTM [43] and CNN-LSTM were evaluated in combination with the above three decoders. LSTM is used in character-level text modeling. The embedding space from the MC and AR models was still inadequate using either encoder (Supplementary Information subsection S1.8). Profile HMM created distinguishable embedding with LSTM, whereas a learning deficiency was observed in combination with CNN-LSTM (Supplementary Information subsection S1.8). Collectively, we concluded that the profile HMM decoder is favorable for motif-dependent embedding. A VAE composed of a CNN encoder and a profile HMM decoder was examined in the following study.

3.1.2 Embedding Split Motifs Using the Profile HMM VAE.

We next tested whether our VAE model could distinguish split motifs. Subsequence co-occurrence at distances is often observed in RNA due to intramolecular base-pairing and internal loop structures [44]. We applied simulation data with a pair of 5-nt split motifs to the VAE (Figure 4). The MC model decoder was used for comparison. Figure 4b shows the results of embedding split motifs. Plots are displayed in three groups: right motif-, left motif-, and both motif-remaining sequences. Profile HMM output sequences related to the motif, whereas the MC model scattered the sequences. We sampled representative profile HMM distributions from each population. Profile HMM visualization shows that the yellow point skips the left motif. The red point skips the right motif, both by allocating a high probability of jumping to the deletion state from the matching state (Figure 4c). Visualization of the purple point shows that the middle of two points has a low probability of skipping either of the motif fragments. The transition probability to skip the left motif a_{M_1,D_2} and the right motif a_{M_10,D_11} for right only-, both-, and left only-motif models were (0.995, 0.000), (0.107, 0.002) and (0.000, 0.987), respectively. Interestingly, the point located between these two motifs has a high probability of including both motifs. These results show that a profile HMM decoder is also applicable for split motifs. Hereafter, we called a VAE with a profile HMM decoder RaptGen.

3.2 Real Data Evaluation with RaptGen

3.2.1 Examining Latent Space Dimension and Model Loss.

We further evaluated RaptGen using SELEX sequence data obtained from our previous study [20]. Because real data are more complex than simulation data, we first investigated the dimensions of the latent space. Raw HT-SELEX data have 30- or 40-nt variable regions and fixed primer regions at both ends. In the present study, we used the variable region to create latent space. We tested up to 12 spatial dimensions and trained the model 50 times on dataset A (Figure 5). The minimum loss was in the four dimensions, and the second-lowest was in the two dimensions. Loss tended to increase as the embedding dimension increased; however, the loss of one-dimensional space was higher than that of the ten-dimensional space. The lower dimension would be favorable for visualization and performing BO would be advantageous, as described in later sections. Therefore, we adopted a two-dimensional space instead of a four-dimensional space for analysis.

3.2.2 Embbedding High-Throughput SELEX Data with RaptGen.

We next subjected two independent high-throughput SELEX datasets, Dataset A and B, to RaptGen. The resulting latent embeddings are shown in Table 1 and the Supplementary Information subsection S1.4. We previously demonstrated that aptamers from Datasets A and B exhibit continuous motifs and split motifs, respectively. As the SELEX experiment sequences are amplified with specific binding motifs, we reasoned that they would form clusters in a latent space based on their motifs. Thus, we used the GMM, which hypothesizes that data consists of a mixture of Gaussian distributions, to classify the distributions. We chose ten different points representing the latent cluster

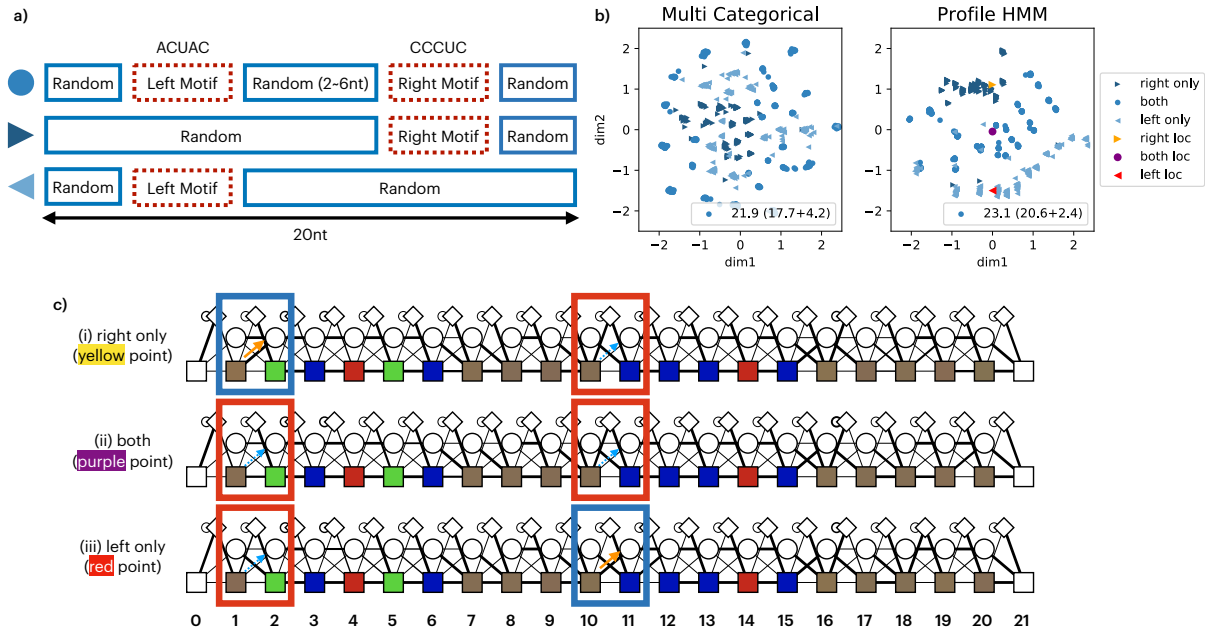


Figure 4. (a) Scheme for paired motifs with simulation data. A set of five nt was used with two- to six-base insertions and extended to 20 bases. Data containing one motif were also generated. (b) Embeddings of split motifs by the VAE. Simulation data generated in (a) were analyzed using the profile HMM VAE. The resulting embedding plot is shown. Plots generated by the VAE with the MC model decoder are also shown for comparison. The circle, right arrowhead, and left arrowhead represent the points from both-, right only-, and left only- motif-containing data, respectively. Yellow, purple, and red indicate representative points for each data point. The yellow and red points show the right and left motif-containing sequences, respectively. (c) Representative profile HMM obtained from the profile HMM VAE. The profile HMM indicated in (b) is shown. The thickness of the line represents the transition probability. The color of the matching state indicates the probability of the emitting nucleotide. A, U, G, and C are green, red, yellow, and blue, respectively. The blue square shows a high probability of skipping motif by moving to the deletion state, whereas the red square highlights a high probability of including the motif.

center of the GMM (Table 1). We observed that sequences with an ambiguous profile HMM such as A-GMM-2, A-GMM5, and B-GMM-0, were embedded near the latent space center. In contrast, the near edge area contained sequences that emit nucleotides preferentially. We also confirmed that similar profiles were embedded in similar areas (Supplementary Information subsection S1.4). These results provide support for the use of RaptGen to analyze high-throughput SELEX data.

3.2.3 Generating Aptamer Candidates from Latent Space.

We attempted to generate the most probable sequence from the profile HMM of each GMM center for activity evaluation. We calculated the model state path with the highest probability and derived the most probable sequence according to the path. When the path included insertion states, we generated up to 256 sequences with no duplication by randomly replacing each insertion state with a single nucleotide and selected a sequence with the highest probability. The resulting reconstituted sequences and their probabilities are shown in Table 1. After connecting with their fixed primer sequences, aptamer RNAs were produced by *in vitro* transcription, and their binding activities were assessed by SPR. Aptamers identified in our previous study were used as positive controls. Although more than half of the candidates were found to have weak or no activity, some sequences such as A-GMM-1, B-GMM-4, and B-GMM-8 had evident binding activity. To determine whether these aptamers exist in the original data, we calculated each sequence's edit distance from the nearest HT-SELEX sequence (Table 2). It should be noted that all candidate sequences were not included in the original SELEX data. Collectively, we concluded that RaptGen enables us to generate aptamers from the latent space and reduces the limitations of

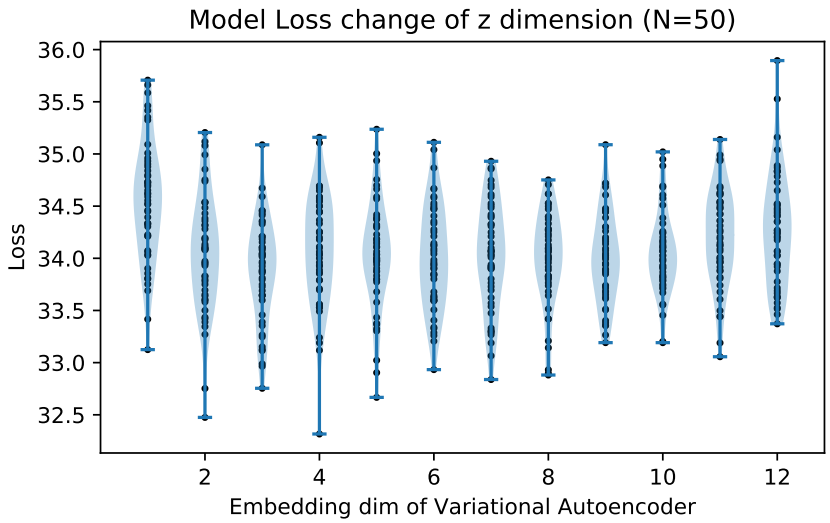


Figure 5. Loss over different latent dimensions. The model was trained using Data A with different dimension numbers. The minimum loss of training was plotted. Data were obtained from 50 runs with random parameter initialization.

working with actual sequence data.

Table 1. The latent embeddings and reconstituted sequences through GMM. Sequences of Data A and B were analyzed by RaptGen. The latent embeddings of Data A and B are shown in the 1st column, where clusters were estimated by GMM. The plot colors indicates the different cluster. Profile HMM were obtained from each center of the gaussian distributions. Their ID, locations in latent space, and Logo views of Profile HMM are listed in the 2nd and 3rd column, respectively. Sequences for activity evaluation were reconstituted from each Profile HMM. The maximum probable sequences were listed in the 4th column.

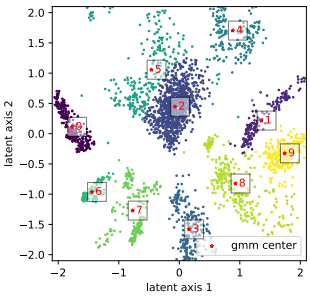
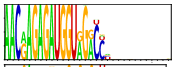
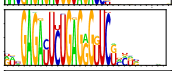
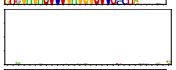
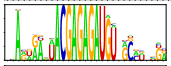
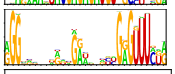
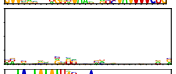
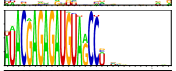
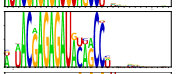
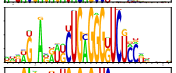
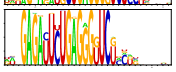
Latent embeddings	ID (dim1, dim2)	Sequence Logo	Maximum probable sequence
	A-GMM-0 (-1.77, 0.11)		AACGAGAGAUGGUAGACCUAUCUUUUAGCC
	A-GMM-1 (1.36, 0.22)		GUAGAGAUUCUGAGGGUUCUCCUGCUAUA
	A-GMM-2 (-0.07, 0.45)		UUUUAUAAAAAAGUGUUUAAAAAAGAUUCA
	A-GMM-3 (0.16, -1.58)		GUAGAAAUUACGAGAGAUGUCGCCUUUGA
	A-GMM-4 (0.88, 1.70)		GGGGGUGCAGUAGAAUUGUCGAGUUUCUG
	A-GMM-5 (-0.46, 1.06)		AAUACCCGGGGUUUUCACACAUAUAAUUCA
	A-GMM-6 (-1.45, -0.96)		AUACGAGAGAUGUAGCCUUUUUUCUGACUU
	A-GMM-7 (-0.77, -1.27)		AGUACGAGAGAUACAGCCUUUUUCCUGCUU
	A-GMM-8 (0.93, -0.82)		GGUAGCAGAUGCUGAGGGGUCUCCUGAUGC
	A-GMM-9 (1.74, -0.33)		GUCGAGAUUCUGAGGGUUCUCCUGUUAACC

Table 1 continued from previous page

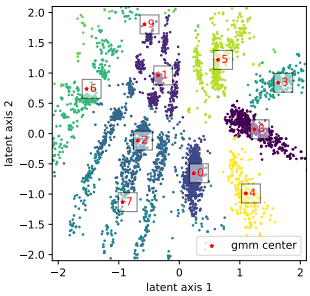
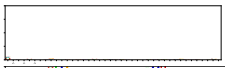
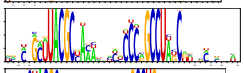
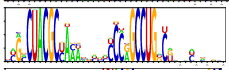
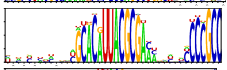
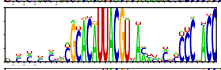
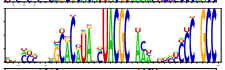
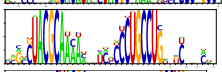
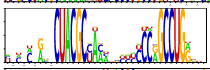
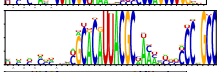
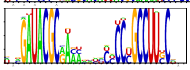
Latent embeddings	ID (dim1, dim2)	Sequence Logo	Maximum probable sequence
	B-GMM-0 (0.24, -0.66)		UACACACCCCCCACCCCCCCCCCGCCCCCCCCCCCCCCAAA
	B-GMM-1 (-0.35, 0.96)		UCUCUGCUUACGCCAAAAUUCCCCCGGCCUAGCUUGGCUCGCUC
	B-GMM-2 (-0.69, -0.12)		ACGCCUACGCCAAAAGCCCCCCAGCCUGGCUUGGCGCGCAC
	B-GMM-3 (1.63, 0.84)		UGCGCGACCGCGCGCACAUUACGCGAAACUCCCCCCCGCC
	B-GMM-4 (1.10, -0.99)		UGCGCGCCCAGCGCACAUUACGUAAAACUCCCCCCUACC
	B-GMM-5 (0.64, 1.22)		UCGCCCUGCGCACAUAACUACGCAAAACUCCCCCCCGCCA
	B-GMM-6 (-1.53, 0.74)		UCGCCUACGCAAAAAACUCCCCCUGCCUGUAUCACUCAC
	B-GMM-7 (-0.94, -1.13)		UGCGCUACUCUACGCUAAACUCCCCCCAGCCUGGAAA
	B-GMM-8 (1.24, 0.07)		UGCGCGCCCGAGCGCACAUUACGCAAAAUCCCCCCUGCC
	B-GMM-9 (-0.58, 1.80)		UGCGAUACGCGAAACGCUCCCCCGCCUUCCUAG

Table 2. The sequences derived by RaptGen from centers of GMM. The log probabilities of the sequences generation from Profile HMM are shown. Aptamer activities were evaluated by SPR assay. Relative activities against respective positive control are shown and the activity higher than the control are shown in bold font. The edit distance to the nearest sequence within the whole sequencing data were calculated.

ID	Log probability	Relative activity	Edit distance
A-GMM-0	-15.8	79.0	4
A-GMM-1	-13.3	**107.1**	4
A-GMM-2	-34.9	-3.6	9
A-GMM-3	-16.7	7.0	3
A-GMM-4	-19.1	15.9	6
A-GMM-5	-30.1	-0.8	7
A-GMM-6	-14.9	42.2	4
A-GMM-7	-17.0	30.9	5
A-GMM-8	-18.7	33.8	5
A-GMM-9	-14.5	74.3	3
B-GMM-0	-47.0	-13.2	15
B-GMM-1	-35.9	21.9	11
B-GMM-2	-29.9	-9.3	10
B-GMM-3	-23.9	74.2	8
B-GMM-4	-23.3	**229.1**	7
B-GMM-5	-26.8	-30.5	9
B-GMM-6	-27.5	-28.5	8
B-GMM-7	-26.4	-7.3	9
B-GMM-8	-24.5	**190.7**	8
B-GMM-9	-23.0	-7.3	8

3.3 RaptGen Application in Aptamer Discovery

3.3.1 Generating Truncated Aptamers using a Short Learning Model.

We proposed further applications of RaptGen for aptamer development. Since the profile HMM can handle variable sequence lengths, learning settings could diverge from the original SELEX library. For example, a decoder model does not require the same length of the random region. We attempted to generate shorter aptamers than SELEX with RaptGen. We introduced a short profile HMM with truncated length by 5 or 10 nt from the original SELEX design. Dataset A was analyzed with a 20-nt and 25-nt model (called L20 and L25), where the initial library was 30 nt. Dataset B was analyzed with a 30-nt and 35-nt model (called L30 and L35), where the initial library was 40 nt. After creating latent space, ten sequences for each length were created in a GMM-dependent manner described above. Figure 6 shows the relative activity of proposed aptamers with their lengths. For dataset A, the 28-nt candidate showed binding activity where the initial library was 30 nt. For dataset B, the 29-nt candidate showed considerable activity compared with the original setting, which was 40 nt. These results suggest that RaptGen can generate a shorter aptamer than the experimentally expected length. We found that sequences with low reconstitution probability tended to have low binding activity and that sequences showing binding activity had relatively high probability (Figure 6). This observation would be helpful for effective candidate selection. We observed a tendency of sequence extension in dataset A-L20 and L25 and dataset B-L35. For instance, in dataset A, 26 nt sequences were generated from the 20 nt RaptGen setting. We speculate that the profile HMM is prone to imitating the original length in some situations. The optimal truncation length was different for each dataset. We did not identify the cause of this difference. Further studies should be performed to determine efficient truncation.

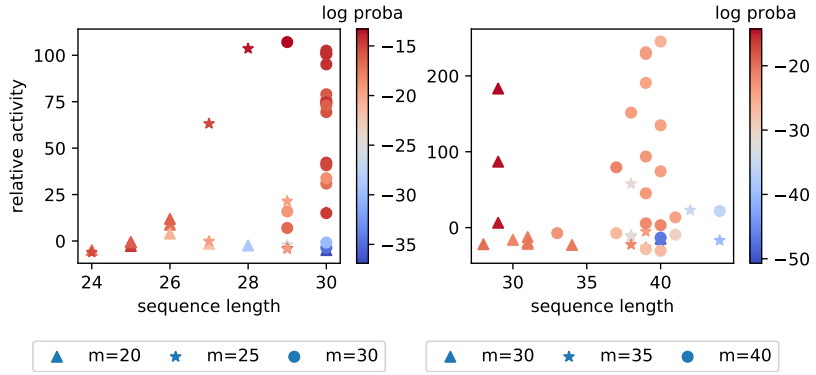


Figure 6. Aptamer length obtained from a short learning model. A profile HMM decoder with a length of 20 nt or 25 nt was used to analyze dataset A, in which the random region of the SELEX library is 30 nt. Similarly, a 30 nt or 35 nt decoder was used to analyze dataset B, in which the random region of the SELEX library is 40 nt. Ten candidate profile HMM were newly obtained by GMM. After reconstitution of the maximum probable sequences, aptamer activities were assessed by SPR. The scatter-plot of relative activities of aptamers and their lengths are shown, including aptamers tested in Table 1. Different markers indicate different lengths of the profile HMM decoder. Colors indicate the log probability of a sequence.

3.3.2 Activity-Guided Aptamer Generation by BO.

In another application of RaptGen, we generated aptamers using activity information. Aptamer derivatives harboring nucleotide mutations should be distributed around the mother sequence in the latent space. To predict effective candidates from the neighboring area of an active aptamer, binding activity distribution should be predicted. We used a BO algorithm for learning an activity distribution. Because the distribution for the BO process is required to be of low dimension, RaptGen is suitable for this strategy. To implement BO, we first embedded activity data in the latent space. The sequences listed in Table 1 were re-converted into the space. Several locations moved from the initial GMM center (Figure 7). We used these re-embedded positions to perform BO. The resulting predicted activity distributions are shown in Figure 7. To propose multiple candidates in parallel, we utilized the local penalization function [45]. Ten profile HMM were proposed and evaluated for their activity. As shown in Figure 7, candidates were generated from the peripheral area of the positive clone. We confirmed that new aptamers incorporated nucleotide substitutions (Table 3). In addition, most of them had binding activity. Similar results were obtained for both datasets A and B. These results indicate that RaptGen can propose aptamer derivatives in an activity-guided manner.

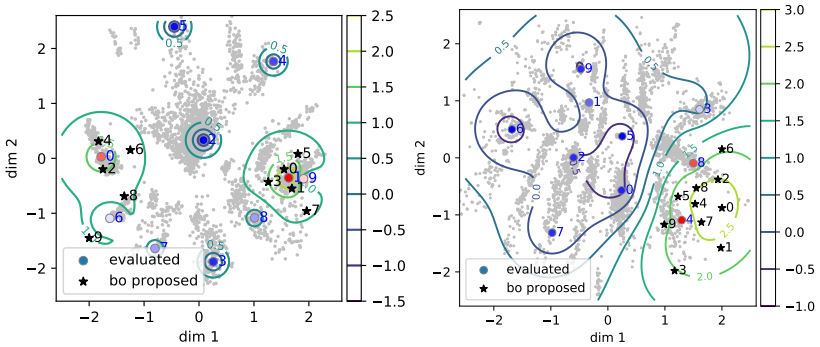


Figure 7. The activity distribution and proposed BO points for data A (left) and data B (right). Binding activity data shown in Table 1 were embedded into latent space. Grey points indicate latent embeddings shown in Table 1. The contour line overlaid on the embeddings indicates the predicted activity level. This is the acquisition function of BO, which is the upper confidence bound of the posterior distribution of the Gaussian process (GP-UCB) [46]. Ten points were proposed by the BO process with a local penalization function. Circles represent the re-embedded position of the GMM centers. Red and blue indicate high and low binding activity, respectively. Stars represent the locations proposed by BO.

Table 3. The top three sequences selected by BO and their nearest tested sequence in the embedded space. Profile HMM obtained by BO were reconstructed into a sequence by deriving the maximum probable sequence. Binding activities of the reconstructed aptamers (sequences) were evaluated by SPR experiment. Top 3 binding aptamers and their activities are shown in alignment view with nearest GMM clones. Hyphen indicates gap and lowercase letter indicates substitutions. Relative activities of A-GMM-1 and B-GMM-4 were redisplayed from Table 1 for comperison.

ID	maximum probable sequence	Relative Activity
A-GMM-1	`GTAGAGATTCTGAGGGTTCTCCTGCT-ATA`	107.1
A-BO-0	`GTAGAGATTCTGAGGGTTCTCCTGtTGAcc`	102.5
A-BO-1	`GTtGAGATTCTGAGGGTTCTCCTGtTGccc`	101.2
A-BO-3	`GTAGAGATTCTGAGGGTTCTCCTGtTGcTA`	100.6
B-GMM-4	`TGCGCGCCC-AGCGCACATTACGTAAAACTCCCCCCTACC`	229.1
B-BO-5	`TGCGCGCCCGAGCGCACATTACGTAAAACTCCCCCCTACC`	245.2
B-BO-4	`TGCGCGCCC-AGCGCACATTACGcAAcACTCCCCCCTgCC`	231.0
B-BO-9	`TGCGCGCCC-AGCGCACATTACGTAAAA-aCCCCCCTACC`	151.6

3.4 Comparison with Other Techniques

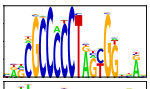
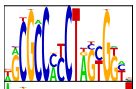
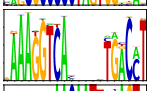
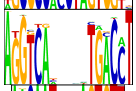
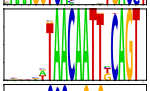
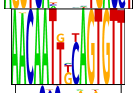
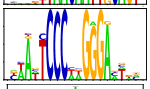
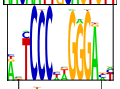
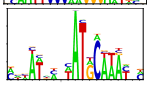
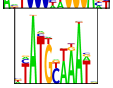
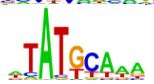
Jolma and colleagues evaluated the binding specificities of transcription factors using SELEX data [8]. In addition, CNN-based research was conducted to classify randomly shuffled sequences and determine the motifs in a dataset [27]. We selected five transcription factors that were presented in these studies. We applied ten GMM distributions in this experiment and identified motifs similar to those from Jolma *et al.* Similarity was based on the edit distance of the most probable sequence from each motif. RaptGen was able to produce sequence motifs consistent with these studies (Table 4). Thus, we could search for sequence motifs obtained by running a CNN. A future study could be performed to search for an appropriate number of distributions using model selection methods such as Akaike Information Criteria (AIC) [47]. Whole GMM motifs and other results are shown in the Supplementary Information Table S3.

3.5 Future Work

The present version of RaptGen does not consider the secondary structure of aptamers. Secondary structure information is critical for identifying active aptamers that [19, 20]. Therefore, including the secondary structure in the sequence probabilistic model would improve RaptGen performance. An alternative model such as profile stochastic context-free grammar (SCFG) [48] will be tested in follow-up studies.

Here, we demonstrated that RaptGen could propose candidates according to activity distribution. According to BO, a sequential construction of posterior distribution would allow us to optimize activity in the latent space. For another instance

Table 4. The motif similarity with previous methods. RaptGen motif was generated as described in Supplementary subsection S1.1. Motif of RaptGen is based on the emission probabilities, motifs of Jolma et al. are based on position frequency matrix (PFM) of multinomial method [15]. The motifs of Alipanahi et al. were based on the PFM, which activated specific motif scanners of the DeepBind model [27], which were derived from deepbind web service `http://tools.genes.toronto.edu/deepbind/`.

Target	RaptGen	Jolma et al.	Alipanahi et al.
CTCF			
NR4A2			
SOX10			
EBF1			
POU2F2			

of BO application, one could set the acquisition function to various indicators other than the binding activity. Therefore, we could generate candidates according to other properties of interest, including inhibitory activity against enzymes or protein-protein interactions. The application of RaptGen for this purpose is promising.

While RaptGen helps visualize and understand sequence motifs, this method requires computational cost due to sequence probability calculation. Compared with the MC model, which can calculate the sequence independently by position, and the AR model that only needs calculation on the previous nucleotides, profile HMM requires calculation on all possible state paths and previous (sub)sequences. The offset calculation cost for MC, AR, and profile HMM is $\mathcal{O}(1)$, $\mathcal{O}(l)$, and $\mathcal{O}(lm)$,respectively, where l is the number of previous characters including itself, and m is the model length of the profile HMM. Profile HMM also needs to calculate the costly logsumexp function many times, leading to a longer training time. Additional studies are necessary to improve these issues.

4 Conclusion

In this study, we developed a novel nucleic acid aptamer discovery method, RaptGen, based on a variational autoencoder and profile hidden Markov model. To our knowledge, this is the first study to utilize a VAE for aptamers. RaptGen converts experimental sequence data into low dimensional space, depending on motif information. Active aptamers were generated from the representation space using a GMM. Aptamers were shorter than the original SELEX aptamers were also generated. Furthermore, we conducted a BO process for derivatizing aptamers based on activity. The creation of efficient embeddings allowed the generation of aptamers *in silico* more efficiently.

5 Funding

This work was supported by JST CREST, Grant Number JPMJCR1881 and JPMJCR21F1, Japan.

6 Acknowledgments

Computation for this study was performed in part on the NIG supercomputer at ROIS National Institute of Genetics. NI and MH thank members of Hamada Laboratory for their valuable comments.

Conflict of Interest Statement

None declared.

S1 Appendix

S1.1 Method to Create Sequence Logo

To create a sequence logo for the given profile HMM, we calculated the most probable path of the profile HMM. The most probable path was iteratively acquired using the following pseudocode.

Algorithm 1 Sequence logo generation

$M[0] \leftarrow 1, D[0] \leftarrow 0$
$I[0] \leftarrow a_{M_0,I_0}$
foreach $i \in [1, m]$ **do**
  $M[i] \leftarrow \max(a_{M_{i-1},M_i} M[i-1], a_{I_{i-1},M_i} I[i-1], a_{D_{i-1},M_i} D[i-1])$
  $I[i] \leftarrow a_{M_i,I_i} M[i]$
  $D[i] \leftarrow \max(a_{M_{i-1},D_i} M[i-1], a_{D_{i-1},D_i} D[i-1])$
end foreach
$M[m+1] \leftarrow \max(a_{M_m,M_{m+1}} M[m], a_{I_m,M_{m+1}} I[m], a_{D_m,M_{m+1}} D[m])$
traceback and acquire most probable path

When calculating the probability of insertion state I, the state's recurrency was ignored because the probability of staying on I recurrently is lower than that of immediately moving to the next state. After the most probable path was achieved, the sequence logo was written according to the emission probability of each state using WebLogo technology [49, 50]. The overall height R_i at state S is defined as

$$R_S = \log_2(|C|) - (H_S + e_n)$$

where $|C|$ is the number of characters (typically 4 for RNA), H_S is the uncertainty at state S, and e_n is the correction factor. The correction factor e_n adjusts the result when there are a few sample sequences. In our setting, we set e_n to 0. The uncertainty H_S is defined as

$$H_S = -\sum_{b \in \{\mathrm{A,U,G,C}\}} e_S(b) \log_2 e_S(b)$$

where b is one of the bases (A, U, G, C) and $e_S(b)$ is the probability of emitting base b from state S. The height of the base at a certain state $h_S(b)$ is defined as

$$h_S(b) = e_S(b) R_S.$$

The sequence logo was written using $h_S(b)$, placing the higher probable base at the bottom and the state path sequence from left to right.

S1.2 Sequence Obtained in the Present Study

The sequences obtained in the present study are shown in Table S1. The ID is named after a rule of dataID, length of the trained model, the method to select the sequence, and the index of the sequence. max_seq indicates that the sequence was the most probable sequence emitted from the most probable state path of the given profile HMM. Relative activity shows % relative binding activities of positive controls assessed by the SPR experiment. The positive control sequences are as follows: A-PC is equal to Data1-11, and B-PC is equal to Data2-1 in our previous report [20] .

Table S1. The sequences obtained through data are shown. Log probability is the probability that the sequence was obtained from a certain point of space. For GMM-, it was obtained in the initial sequence selection, and for BO-, it was obtained in the next candidate selection procedure.

ID	maximum probable sequences	length	log probability	relative activity
A-PC	GCGGAGAUUCUGAGGGUUCUCCUGUUCCAG	30	-	(100)
A-L20-GMM-0	CUCGAGAUUCUGAGGGUUCUGCAUA	25	-13.8	-0.7
A-L20-GMM-1	AAGAAAUUAGAUACUAGUAAAUACGACAUA	30	-36.9	-4.9
A-L20-GMM-2	UGUGCGGAUAUGCUGAGUUUCUGC	24	-17.9	-4.9
A-L20-GMM-3	AACGAGAGAUACAACCUGUGCGUGCC	26	-17.0	8.8
A-L20-GMM-4	AUCGAGAGAUGGUAGACCCUGUGCG	25	-14.6	-2.7
A-L20-GMM-5	CAAAGACGUGCAAGCCUCGUCUACGGU	27	-21.8	-1.5
A-L20-GMM-6	UCUGAGGGUCCUGACAACAGAAACA	25	-16.2	-0.7
A-L20-GMM-7	UACGAGAGAUGUAGCCUGUAUGUGCU	26	-16.3	11.9
A-L20-GMM-8	CAGUAAGGUUUCGACACUCUUCUACUUA	28	-28.9	-2.4
A-L20-GMM-9	AACGAGAGAUGUUGCCUGCGUGUUGC	26	-20.8	4.0
A-L25-GMM-0	CAUCAAUAAUAAGAAUAACAAAAUUUCAAA	30	-35.5	-1.1
A-L25-GMM-1	CAACUACGAGAGAUGUAGCCUGGA	24	-15.4	-6.1
A-L25-GMM-2	GUAGAGAUUCUGAGGGUUCUCCCGCUCC	28	-13.7	103.6
A-L25-GMM-3	GGCGGUGAUGUAGAAACGGUUGAGGUUAA	29	-25.0	-2.1
A-L25-GMM-4	AUACGAGAGAUGUAGCCUGUGUCGUAGAA	29	-14.7	-4.0
A-L25-GMM-5	AACGAGAGAUGGUAGACCGUUGUGGAU	27	-15.3	63.2
A-L25-GMM-6	CACGCGGGUUUCACACUCUAUGAAUGA	27	-19.5	-0.1
A-L25-GMM-7	CGUAGGGAAUCUGAGGGGUCUCCCGCCCU	29	-19.7	21.5
A-L25-GMM-8	AACGGUAACACGUGCAAGCCUGUUAUUAU	29	-19.3	-4.3
A-L25-GMM-9	AUUACGAGAGAUACAGCCUAUAUCGUAGCA	30	-15.4	15.4
A-GMM-0	AACGAGAGAUGGUAGACCUAUCUUUUAGCC	30	-15.8	79.0
A-GMM-1	GUAGAGAUUCUGAGGGUUCUCCUGCUAUA	29	-13.3	107.1
A-GMM-2	UUUUAUAAAAAAGUGUUUAAAAAAGAUUCA	30	-34.9	-3.6
A-GMM-3	GUAGAAAUUACGAGAGAUGUCGCCUUUGA	29	-16.7	7.0
A-GMM-4	GGGGGUGCAGUAGAAUUGUCGAGUUUCUG	29	-19.1	15.9
A-GMM-5	AAUACCCGGGGUUUUCACACAUAUAAUUCA	30	-30.1	-0.8
A-GMM-6	AUACGAGAGAUGUAGCCUUUUUUCUGACUU	30	-14.9	42.2
A-GMM-7	AGUACGAGAGAUACAGCCUUUUUCCUGCUU	30	-17.0	30.9
A-GMM-8	GGUAGCAGAUGCUGAGGGGUCUCCUGAUGC	30	-18.7	33.8
A-GMM-9	GUCGAGAUUCUGAGGGUUCUCCUGUUAACC	30	-14.5	74.3
A-BO-0	GUAGAGAUUCUGAGGGUUCUCCUGUUGACC	30	-14.4	102.5
A-BO-1	GUUGAGAUUCUGAGGGUUCUCCUGUUGCCC	30	-14.4	101.2
A-BO-2	AACAAGAGAUGGUAGACCUAUCUCUUACCC	30	-15.6	69.4
A-BO-3	GUAGAGAUUCUGAGGGUUCUCCUGUUGCUA	30	-14.3	100.6
A-BO-4	AACGAGAGAUGGUAGACCUAUCUUUUAGCC	30	-16.2	76.0
A-BO-5	GUCGAGAUUCUGAGGGUUCUCCUGGUGACC	30	-14.4	74.6
A-BO-6	AACGAGAGAUGGUAGACCUAUUUUUUAGUC	30	-16.2	73.1
A-BO-7	AAUGAGAUUCUGAGGGGUCUCCUGUUGCCA	30	-14.4	95.1
A-BO-8	AUACGAGAGAUGUAGCCUUUUUUCUUACUU	30	-15.2	40.8
A-BO-9	AUACGAGAGAUGUAGCCUUUUUACCGACCU	30	-14.7	15.0

Table S1 continued from the previous page

ID	maximum probable sequences	length	log probability	relative activity
B-PC	GCUGUGUCUACGUCCGGAUUGGGGACCUGCACGGCCCAUG	40	-	(100)
B-L30-GMM-0	ACUCACAUUACGUAAAAAUCGCCCCUACC	29	-14.9	6.4
B-L30-GMM-1	UGGAUACGCAAAAGCUCCCCCUGCCUUACA	30	-21.6	-16.1
B-L30-GMM-2	UAAAACAGCUGCCCCCCCCCCAUCUGACCCGACGACCAAC	40	-50.7	-14.6
B-L30-GMM-3	UACGACCAGCUACGCAACAGUUCUCCCUGCCUGA	34	-20.7	-22.9
B-L30-GMM-4	UUCGCUACGCGAAAGUUCCCCCCGCCUGGCG	31	-21.0	-12.4
B-L30-GMM-5	UUCGCUACGCUAAAGCUCUCCCAGCCUGGCG	31	-21.7	-21.6
B-L30-GMM-6	ACUCACAUUACGCAAAACUCGCCCCUGCC	29	-14.7	86.9
B-L30-GMM-7	UCUACGUCAAACUCGCCCCGACCUGGCG	28	-20.0	-21.9
B-L30-GMM-8	ACACACAUUACGCGAAACUCGCCCCCGCC	29	-14.3	183.3
B-L30-GMM-9	UUCGCUACGCAAAAGUUCCCCCUGCCUGGCG	31	-20.5	-19.7
B-L35-GMM-0	UGCUCGACACUACGCAACACUUCCCCCUGCCUGACAUA	38	-31.0	-12.1
B-L35-GMM-1	UCCGCACGCCAGCGCACAUUACGUAAAGAUCGCCCUACC	39	-23.9	-5.2
B-L35-GMM-2	UGCACGACGCUACGCCAAACUCCCCCCGGCCUGAUAAA	38	-31.5	57.7
B-L35-GMM-3	UAACACAGCCCCACCCCUCGGACGACCGGAGGAAUAAAAA	40	-44.8	-14.9
B-L35-GMM-4	UGCUCUAGCUACGCGAAAAUCCCCCCCGCCUGCAUGCGCACA	42	-35.2	23.0
B-L35-GMM-5	UCCGCCCGCCAGCGCACAUUACGCAAAGAUCGCCCUGCC	39	-24.8	-25.7
B-L35-GMM-6	UUCGCUAGCGUACGCAAAACUCCCCCCUGCCUUGCAUGCGCUUA	44	-40.3	-16.9
B-L35-GMM-7	UACCACGCCCGCGCACAUUACGCGAAGAUCGCCCCGCC	38	-21.9	-22.4
B-L35-GMM-8	UCCGCCAACCCCCCCCCUCCCCCACCCCCCCACCUCCAAA	40	-47.5	-22.5
B-L35-GMM-9	UGCGCGUCACUACGCUAAAAUCCCCCCAGCCUGACAAA	38	-31.9	-10.0
B-GMM-0	UACACACCCCCCACCCCCCCCCCGCCCCCCCCCCCCCAAA	40	-47.0	-13.2
B-GMM-1	UCUCUGCUUACGCCAAAAUUCCCCCGGCCUAGCUUGGCUCGCUC	44	-35.9	21.9
B-GMM-2	ACGCCUACGCCAAAAGCCCCCCAGCCUGGCUUGGCGCGCAC	41	-29.9	-9.3
B-GMM-3	UGCGCGACCGCGCGCACAUUACGCGAAACUCCCCCCCGCC	40	-23.9	74.2
B-GMM-4	UGCGCGCCCAGCGCACAUUACGUAAAACUCCCCCCUACC	39	-23.3	229.1
B-GMM-5	UCGCCCUGCGCACAUAACUACGCAAAACUCCCCCCCGCCA	40	-26.8	-30.5
B-GMM-6	UCGCCUACGCAAAAAACUCCCCCUGCCUGUAUCACUCAC	39	-27.5	-28.5
B-GMM-7	UGCGCUACUCUACGCUAAACUCCCCCCAGCCUGGAAA	37	-26.4	-7.3
B-GMM-8	UGCGCGCCCGAGCGCACAUUACGCAAAAUCCCCCCUGCC	39	-24.5	190.7
B-GMM-9	UGCGAUACGCGAAACGCUCCCCCGCCUUCCUAG	33	-23.0	-7.3
B-BO-0	UGCGCGAGCCGCGCCAUUACGCAACACUCGCCCCCUGCC	39	-22.0	5.6
B-BO-1	UGCGCGAGGCCGCGCCAUUACGUAACACUCGACCCCUACC	40	-21.8	3.2
B-BO-2	UGCGCGACCCGCGCCAUUACGCAACACCCCCCCUGCC	37	-21.6	79.4
B-BO-3	UGCGCGAGCCGCGCACAUUACGUAAAAAAUCGACCCCUACC	41	-24.3	13.5
B-BO-4	UGCGCGCCCAGCGCACAUUACGCAACACUCCCCCCUGCC	39	-23.8	231.0
B-BO-5	UGCGCGCCCGAGCGCACAUUACGUAAAACUCCCCCCUACC	40	-25.8	245.2
B-BO-6	UGCGCGACCGCGCGCACAUUACGCAACACCCCCCCUGCC	39	-23.4	93.7
B-BO-7	UGCGCGAGCAGCGCCAUUACGUAACACUCGCCCCCUACC	39	-23.3	45.2
B-BO-8	UGCGCGCCCGCGCGCACAUUACGCAACACUCCCCCCUGCC	40	-24.2	134.9
B-BO-9	UGCGCGCCCAGCGCACAUUACGUAAAAACCCCCCUACC	38	-23.7	151.6

S1.3 Statistics of the Dataset

The statistics of the datasets are shown in Table S2. The column names are as follows:

- ID : The ID named after the rule; {indicator of the dataset}-{round of the SELEX}.
- raw reads : The number of reads acquired in the sequencing procedure.
- no filter unique : The number of the unique sequences with no filtration.
- adapter match : The number of sequences that match the forward- and reverse-adapter.
- designed length : The number of sequences that match the adapters and also the length of the sequence matches the experimentally designed length.
- filtered unique : The number of unique sequences that passed both adapter filtering and design length filtering.
- > 1 : The number of filtered unique sequences that read more than once.
- $U(T)$: The unique ratio defined in the main paper. The ratio was calculated using filter-passed sequences.
- $\Delta U(T)$: The difference in unique ratio with the previous round.

Note that the data with the lowest $U(T)$, which holds $U(T) > 0.5$, were used.

Table S2. Statistics of the datasets used in our research.

ID	raw reads	no filter unique	adapter match	designed length	filtered unique	> 1	$U(T)$	$\Delta U(T)$
A-0R	162003	159606	114717	93280	91899	1353	0.985	-
A-1R	91610	90225	72675	58555	57595	929	0.984	0.002
A-2R	45431	44829	36696	29296	28856	424	0.985	-0.001
A-3R	91441	90140	73054	58235	57349	857	0.985	0.000
A-4R	80864	65532	64503	51536	38513	3043	**0.747**	0.237
A-5R	108428	48575	86862	70785	20482	3760	0.289	0.458
A-6R	98237	25981	80801	67871	7750	2180	0.114	0.175
A-7R	49469	12565	40306	33101	3117	1118	0.094	0.020
A-8R	113137	26409	81177	67153	3312	1263	0.049	0.045
B-3R	146505	141174	102126	74454	72395	1874	0.972	-
B-4R	121185	85170	83310	58405	31358	4510	**0.537**	0.435
B-5R	116917	57404	83869	57955	13587	3375	0.234	0.302
B-6R	82488	37867	57827	34446	6064	1704	0.176	0.058

S1.4 Sequence Logo Map

We created a map of sequence logos for the two sets of data acquired using the sequence logo creation method, as mentioned in subsection S1.1. This sequence logo is a visualization of the profile HMM at a point equally divided from -2.5 to 2.5 on each axis of the two-dimensional latent space. The sequence logo map for data A is shown in Figure S1, and for data B is shown in Figure S2.

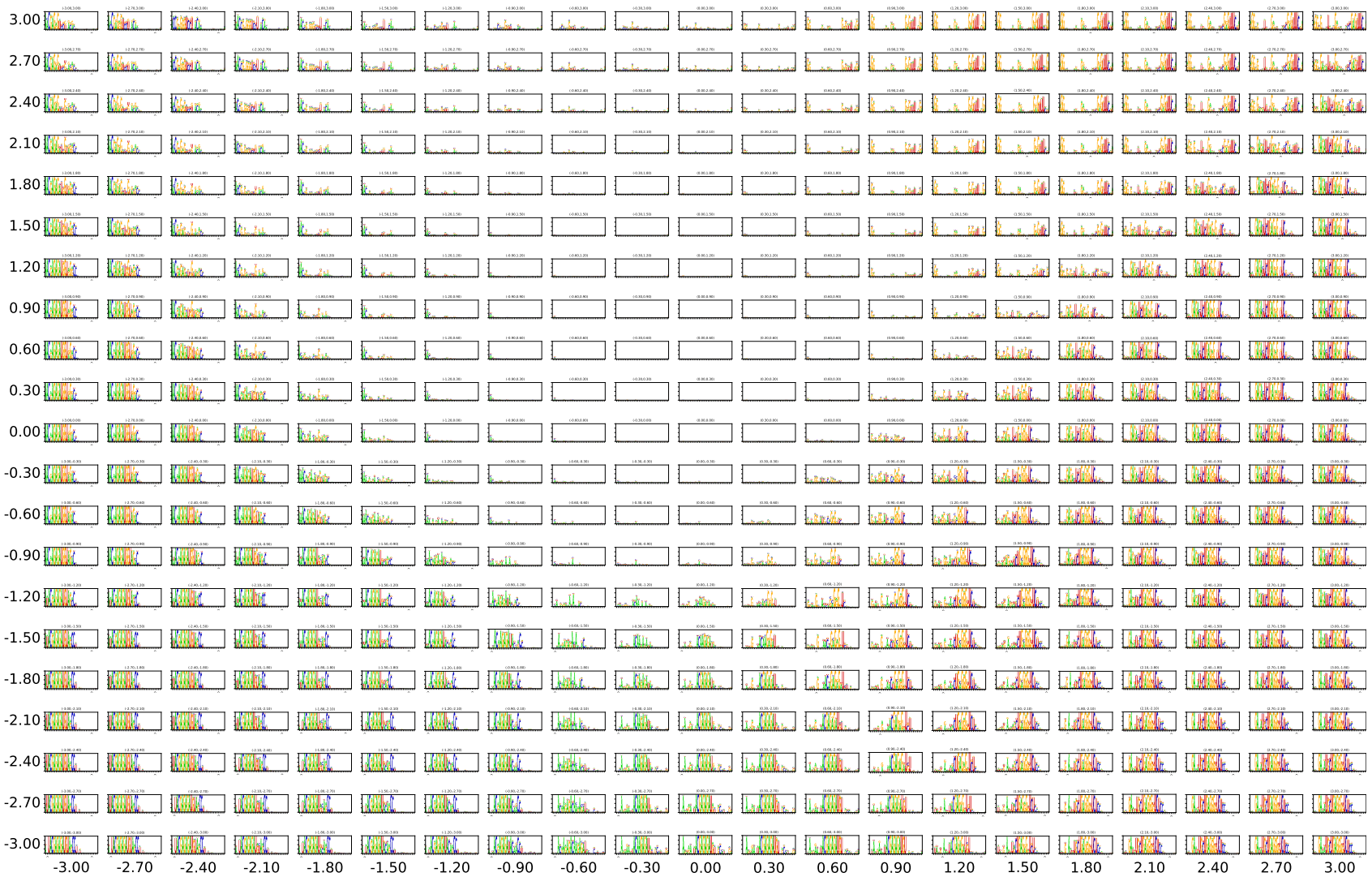


Figure S1. The sequence logo map for data A. The continuous motif indicated that the sequences in the data had the preference for specific subsequences. The result was partly consistent with our previous research [20] that implied `AGAGAUGGUA` was the truncated motif.

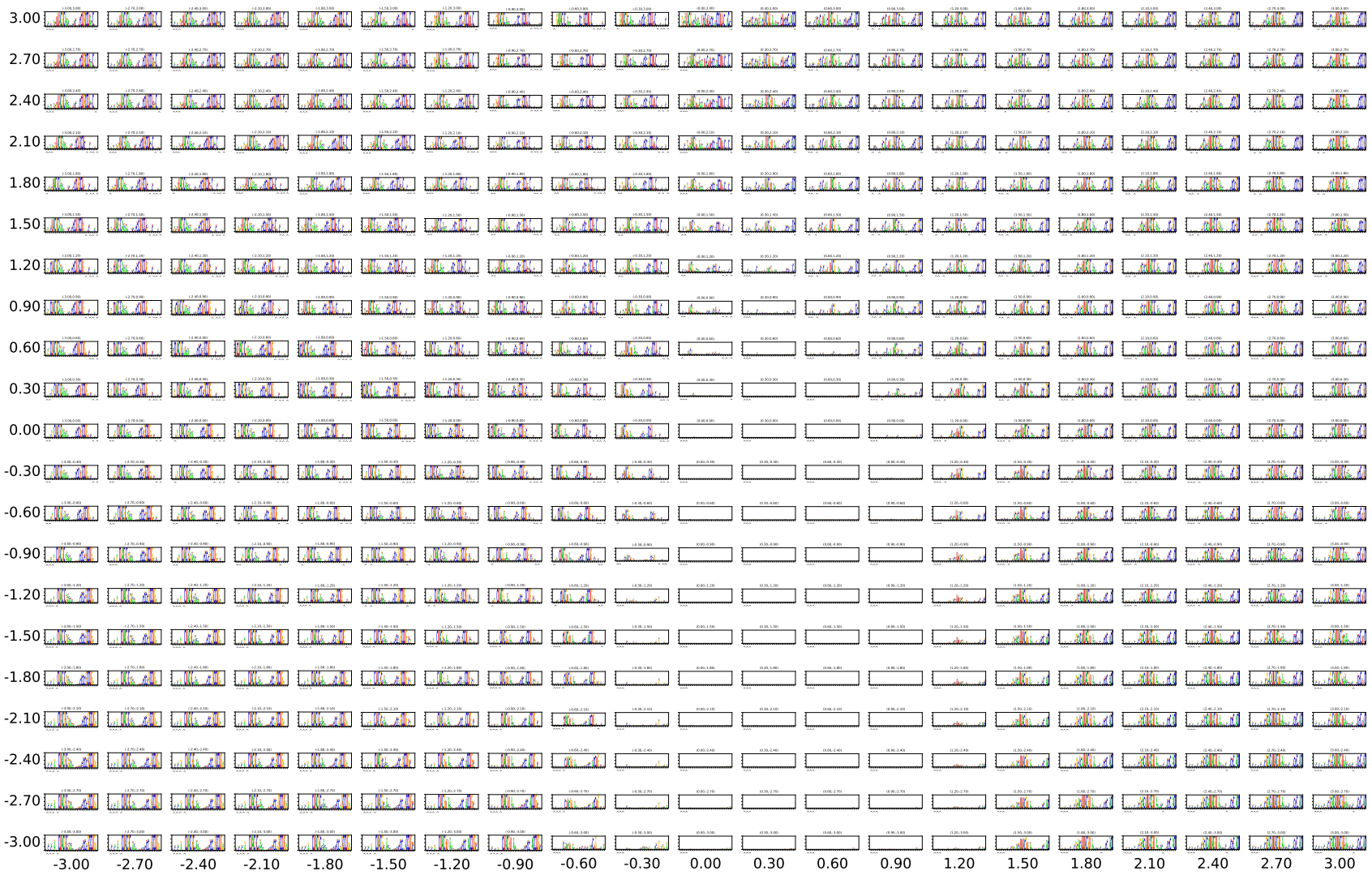


Figure S2. The sequence logo map for data B. The logos indicate that the Profile HMM at the center of the latent space had no preference to emit sequences while the outer surrounding points had captured split motifs.

S1.5 Structure of RaptGen

The structure of RaptGen is shown in Figure S3. The RaptGen consists of a convolutional neural network (CNN)-based encoder and a profile HMM for decoder distribution. We further tried recurrent neural networks with long short-term memory (LSTM) and CNN-LSTM, a network of CNN followed by LSTM. The CNN utilized skip-connection [51], which enables deeper layers to learn appropriately. The models implemented in this study are available at `https://github.com/hmdlab/raptgen`.

S1.6 Encoders

S1.6.1 Convolutional neural network

A CNN captures sequential motifs that are aligned in certain positions. The encoder CNN is a network that first embeds each character into a 32-dimensional vector. Then, the six layers of the skip-connection layer capture the interactions. Finally, the max-pooling layer outputs the resulting feature vector. The skip-connection layer is a combination of a convolutional layer, batch normalization [52], and leaky rectified linear unit (leaky ReLU) [53]. The structure of the skip-connection layer is

$$\begin{aligned}
\mathbf{x}_1 &= \mathrm{Conv}_1(\sigma(\mathrm{BN}(\mathbf{x}_{\mathrm{in}}))) \\
\mathbf{x}_2 &= \mathrm{Conv}_2(\sigma(\mathrm{BN}(\mathbf{x}_1))) \\
\mathbf{x}_3 &= \mathrm{Conv}_3(\sigma(\mathrm{BN}(\mathbf{x}_2))) \\
\mathbf{x}_{out} &= \sigma(\mathbf{x}_{\mathrm{in}} + \mathbf{x}_3)
\end{aligned}$$

where $\mathbf{x}_{\mathrm{in}}$ is the input vector, $\mathrm{BN}(\cdot)$ is the batch normalization layer, and $\sigma(\cdot)$ is the leaky ReLU layer. Convolutional layer Conv_1, Conv_2, and Conv_3 transforms vector dimensions from 32 to 64, 64 to 64, and 64 to 32, respectively. All convolutional layers had the same kernel size of 7 and a zero-padding length of 3 on both edges. Finally, max-pooling, taking the maximum value along the sequence, was performed, resulting in a 32-dimensional feature vector. An encoder CNN was used in the RaptGen architecture.

S1.6.2 Recurrent neural network with long short-term memory

A recurrent neural network can consider the context of the input sequence. LSTM is an artifact that can handle the gradient vanishing problem, the hardness to learn long sequential data [43]. We used bidirectional LSTM for encoding sequences. The sequence was first embedded into a 32-dimensional vector, similar to the CNN encoder, and then it was calculated through a 16-dimensional hidden vector in each direction. The final hidden vector for each direction was concatenated into a 32-dimensional vector, and this was used for the feature vector.

S1.6.3 CNN-LSTM

The convolutional layer and recurrent layer are used in combination to consider both fixed-length and long-distance interactions. The CNN was almost the same as described in the previous section, with the difference that the final max-pooling was removed. LSTM treated this feature vector as sequence embedding as described in subsubsection S1.6.2.

S1.7 Decoders

S1.7.1 Multi categorical model

The multi-categorical model gives the probability to each position of the fixed-length sequence. The output of the model is

$$\begin{aligned}
\mathbf{x}_1 &= \sigma(\mathrm{BN}(\mathrm{FC}_{\mathrm{D},32}(\mathbf{z}))) \\
\mathbf{x}_2 &= \sigma(\mathrm{BN}(\mathrm{FC}_{32,64}(\mathbf{x}_1))) \\
\mathbf{x}_3 &= \sigma(\mathrm{BN}(\mathrm{FC}_{64,32}(\mathbf{x}_2))) \\
\mathbf{x}_4 &= \mathrm{FC}_{32,32\times L}(\mathbf{x}_1 + \mathbf{x}_3)
\end{aligned}$$

Where $\mathbf{z}$ is the sampled vector and $\mathrm{FC}_{J,K}$ is a fully connected layer, which is defined as $\mathrm{FC}_{J,K}(\mathbf{x}) = W_{\mathrm{J,K}}^{\mathrm{T}}\mathbf{x}$ with the learnable parameter $W_{\mathrm{J,K}} \in \mathcal{R}^{\mathrm{J}\times\mathrm{K}}$. To interpret the interactions of each other, the embedding parameter is calculated using the transposed convolution function [54], which is generally the opposite of a convolutional function. The final output $\mathbf{x}_{\mathrm{out}}$ is

$$\begin{aligned}
\mathbf{x}_5 &= \mathrm{Trans_Conv}_{32,32}(\sigma(\mathrm{BN}(\mathbf{x}_4))) \\
\mathbf{x}_6 &= \mathrm{Trans_Conv}_{32,32}(\sigma(\mathrm{BN}(\mathbf{x}_5))) \\
\mathbf{x}_{\mathrm{out}} &= \sigma(\mathrm{Trans_Conv}_{32,32}(\sigma(\mathrm{BN}(\mathbf{x}_6))))
\end{aligned}$$

where $\mathrm{Trans_Conv_{J,K}}(\cdot)$ is the transposed convolutional function with the trainable parameters of the input dimension J and output dimension K. All transposed convolutional layers have the same kernel size of 7 and the same padding length of 3.

S1.7.2 Autoregressive model

To run the autoregressive model, we used a gated recurrent unit (GRU), which is a simplified version of LSTM [55]. The GRU is calculated as follows:

$$\begin{aligned} z_t &= \sigma_g\left(W_z x_t + U_z h_{t-1} + b_z\right) \\ r_t &= \sigma_g\left(W_r x_t + U_r h_{t-1} + b_r\right) \\ h_t &= (1 - z_t) \circ h_{t-1} + z_t \circ \sigma_h\left(W_h x_t + U_h\left(r_t \circ h_{t-1}\right) + b_h\right) \end{aligned}$$

where t is time, x_t is the input vector, h_t is the output vector, z_t is the update gate vector, r_t is the initializing gate vector, and W, U, and b are parameter vectors. The probability of output sequence $p(\mathbf{x}) = \prod_{i=1}^{L} p(x_i \mid x_{0:i-1}, \mathbf{z})$ is calculated by

$$\begin{aligned} \mathbf{x}_1 &= \sigma(\mathrm{BN}(\mathrm{FC}_{\mathrm{D},32}(\mathbf{z}))) \\ \mathbf{x}_2 &= \sigma(\mathrm{BN}(\mathrm{FC}_{32,64}(\mathbf{x}_1))) \\ \mathbf{x}_3 &= \sigma(\mathrm{BN}(\mathrm{FC}_{64,32}(\mathbf{x}_2))) \\ \mathbf{x}_4 &= \mathrm{GRU}(\mathbf{x}_{\mathrm{in}}, \mathbf{x}_1 + \mathbf{x}_3) \\ \mathbf{x}_{\mathrm{out}} &= \sigma(\mathrm{FC}_{32,32}(\mathbf{x}_4)) \end{aligned}$$

where $\mathrm{GRU}(\mathbf{x}, h_0)$ is defined as a GRU function with input vector $\mathbf{x}$ of length L and initial hidden vector h_0, which outputs h_L.

S1.7.3 Profile hidden Markov model

The profile HMM is described in Figure S3b. The embedded sequence is a D-dimensional vector, which is transformed into 32 dimensions by a fully connected (FC) layer. After the vector is rectified by leaky ReLU, the vector is transformed into a certain shape to fit the parameters of Profile HMM. For each parameter, FC, leaky ReLU, FC, and reshape procedure is performed.

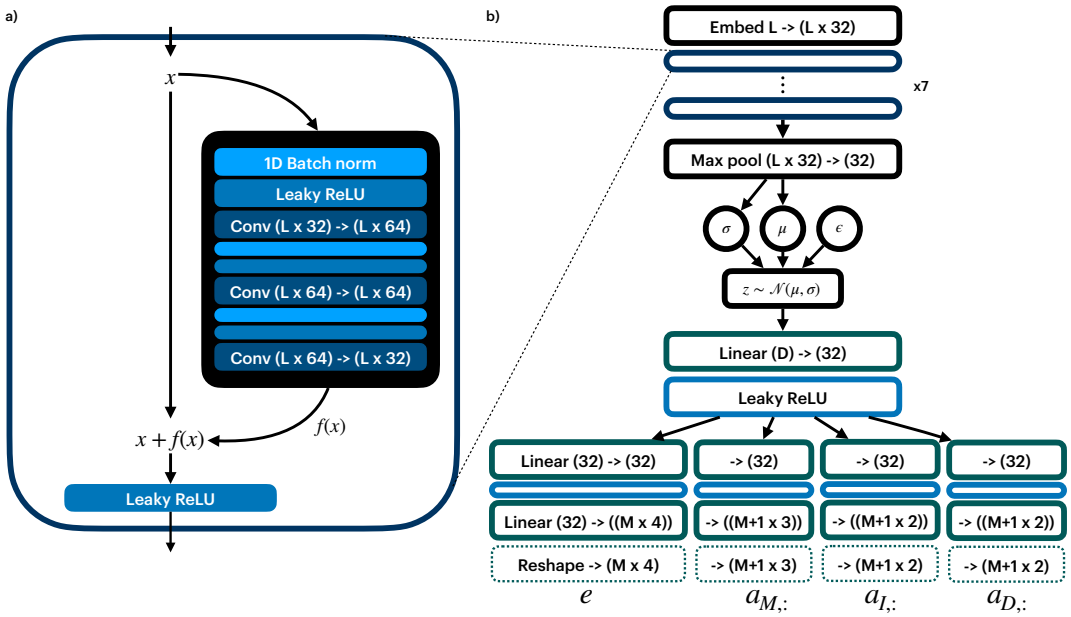


Figure S3. (a) The skip-connection layer. The input feature vector is first passed through 64 hidden layers, then through 32 layers, and added to the original vector. It then passes through the Leaky ReLU normalization layer and produces output. (b) Overall architecture of RaptGen. The input sequence is initially embedded into 32 feature vector and goes through skip-connection layers. After the latent mean and log-variance are calculated with the fully connected layer, which is written as "Linear," the latent variable is sampled and calculated to fit the profile HMM parameter shapes.

S1.8 The Embeddings of Different Encoder and Decoder Combinations

The embeddings are shown in Figure S4. The embedding of the multi-categorical probabilistic model tends to place sequences near the same motif; however, the nearest surrounding sequence is not from the same motifs. Although the autoregressive model has a lower loss, it tends to have an unsplit latent space.

S1.9 Comparison with Other Experiments

In 2013, Jolma et al. conducted a large-scale experiment to identify transcription factor binding sites using SELEX experiments [8]. We utilized the data from that research and estimated the latent embedding, and checked whether the derived motif logo was consistent. We selected five targets whose logo was mentioned in the research of DeepBind [27]. Table S3 shows learned embedding spaces and the sequence logo of the GMM center trained on 10 components. Although the embeddings did not clearly split into 10 areas, the Profile HMM logo was consistent with previously determined motifs. Logo images of previous research were downloaded from the CisBP database [56]. The motif learned by Deepbind with the top three weights is also shown.

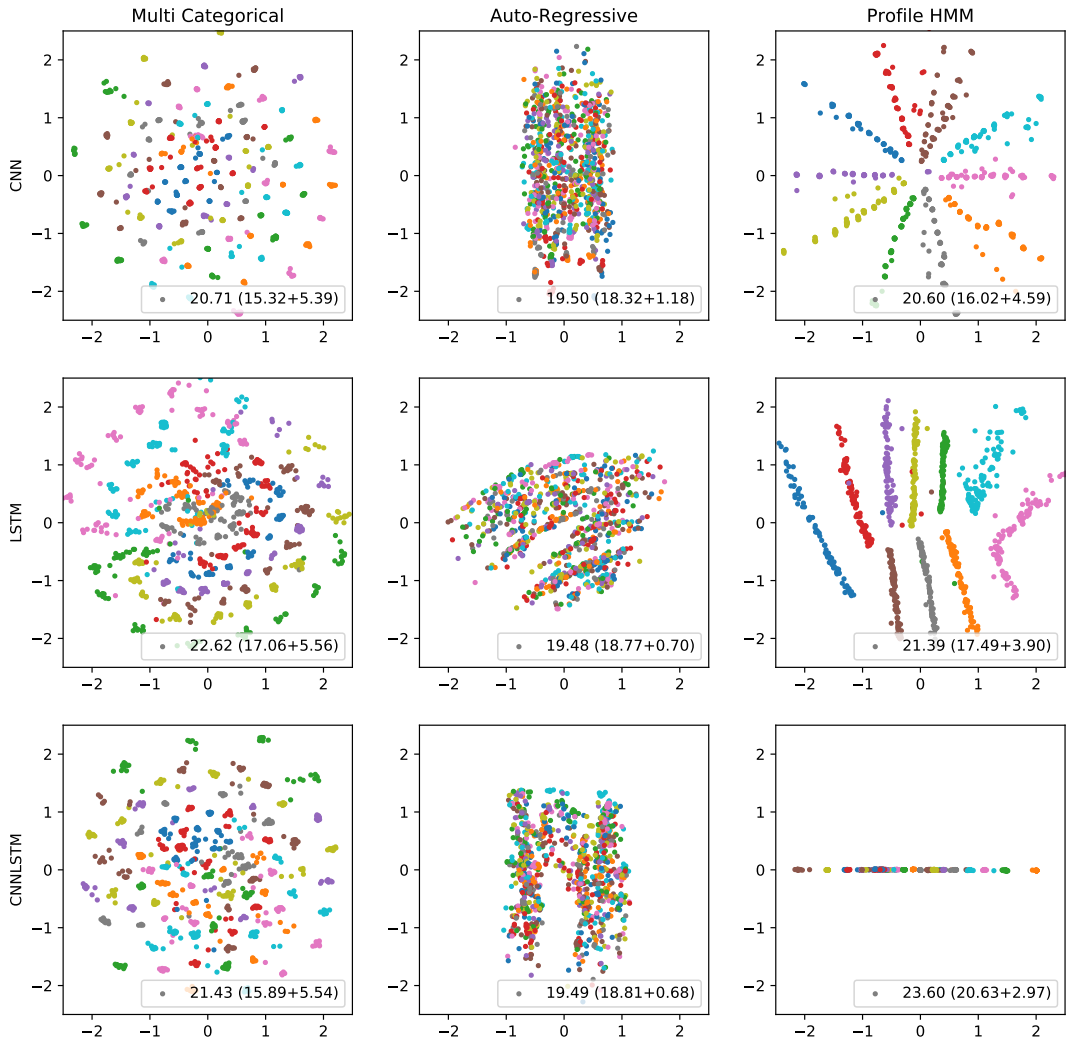


Figure S4. Embeddings of different encoders and decoders. The minimum loss is written at the bottom-right corner. The reconstruction error is to the left, and the regularization error is to the right in the braces. The color represents the motif that each sequence contains.

Table S3. Sequences taken from the SELEX experiment in a previous study [8] are shown. Position interdependent motifs (CTCF, NR4A2), secondary motifs(Sox10, Pou2f2), and motif suggesting potential co-factors (EBF1) are presented.

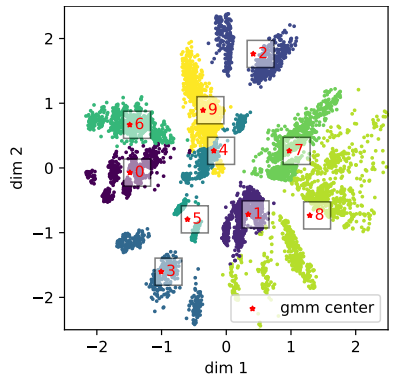
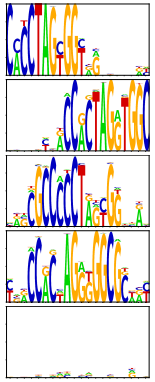
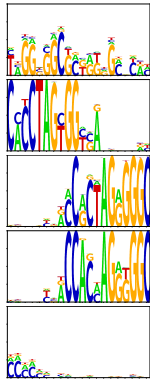
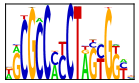
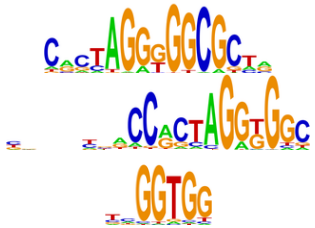
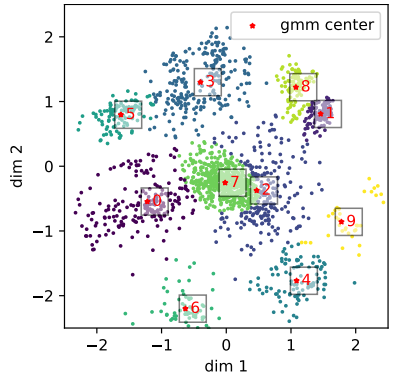
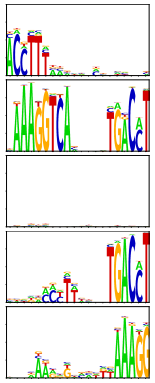
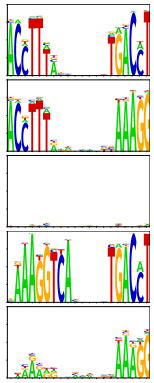
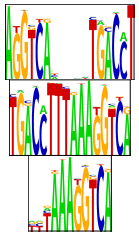
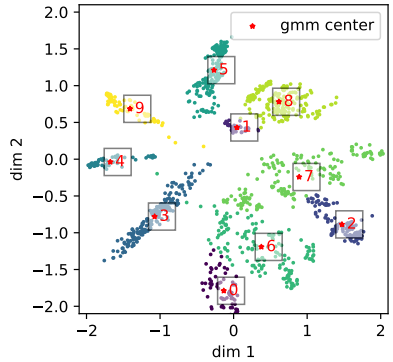
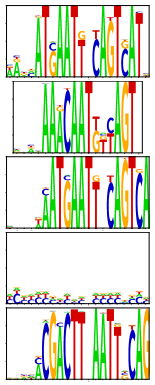
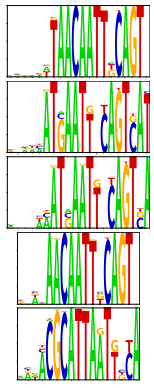
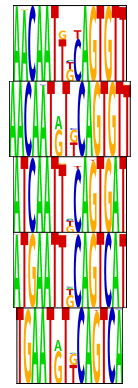
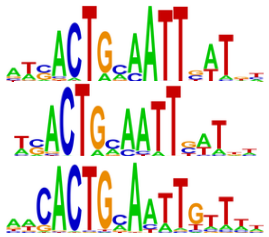
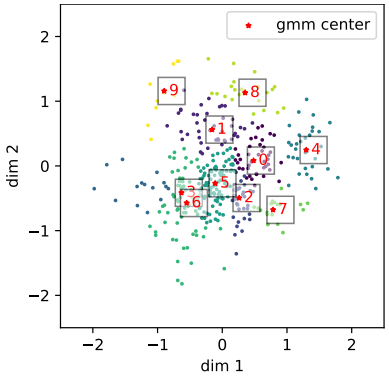
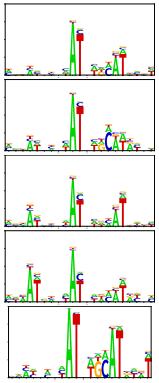
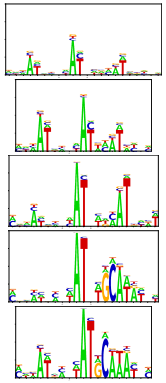
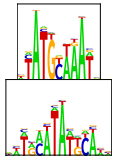
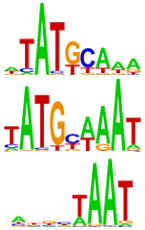
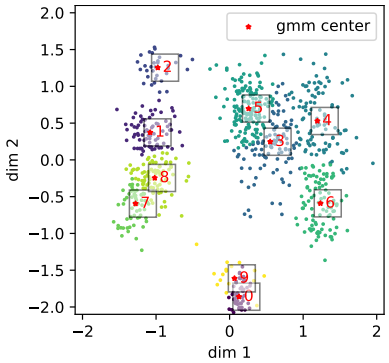
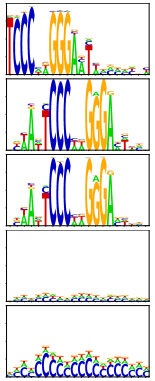
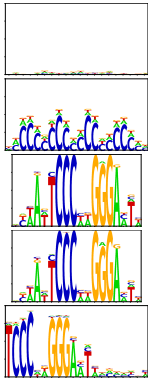
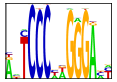
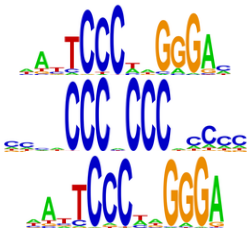
target	embeddings	idx	seq logo	idx	seq logo	jolma et al.	Alipanahi et al.
CTCF		0 1 2 3 4		5 6 7 8 9			
NR4A2		0 1 2 3 4		5 6 7 8 9			
SOX10		0 1 2 3 4		5 6 7 8 9			

Table S3 continued from the previous page

target	embeddings	idx	seq logo	idx	seq logo	Jolma et al.	Alipanahi et al.
POU2F2		0 1 2 3 4		5 6 7 8 9			
EBF1		0 1 2 3 4		5 6 7 8 9			

References

1. Shuaijian Ni, Zhenjian Zhuo, Yufei Pan, Yuanyuan Yu, Fangfei Li, Jin Liu, Luyao Wang, Xiaoqiu Wu, Dijie Li, Youyang Wan, et al. Recent progress in aptamer discoveries and modifications for therapeutic applications. *ACS Applied Materials & Interfaces*, 2020.
2. Tatsuo Adachi and Yoshikazu Nakamura. Aptamers: A review of their chemical properties and modifications for therapeutic application. *Molecules*, 24(23):4229, 2019.
3. Shiping Song, Lihua Wang, Jiang Li, Chunhai Fan, and Jianlong Zhao. Aptamer-based biosensors. *TrAC Trends in Analytical Chemistry*, 27(2):108–117, 2008.
4. Wenhu Zhou, Po-Jung Jimmy Huang, Jinsong Ding, and Juewen Liu. Aptamer-based biosensors for biomedical diagnostics. *Analyst*, 139(11):2627–2640, 2014.
5. Eyetech Study Group et al. Preclinical and phase 1a clinical evaluation of an anti-vegf pegylated aptamer (eye001) for the treatment of exudative age-related macular degeneration. *Retina*, 22(2):143–152, 2002.
6. Jerzy Ciesiolka, Jessica Gorski, and Michael Yarus. Selection of an rna domain that binds zn2+. *Rna*, 1(5):538–550, 1995.
7. Sara Tombelli, Maria Minunni, Ettore Luzi, and Marco Mascini. Aptamer-based biosensors for the detection of hiv-1 tat protein. *Bioelectrochemistry (Amsterdam, Netherlands)*, 67:135–41, 11 2005.
8. Arttu Jolma, Jian Yan, Thomas Whitington, Jarkko Toivonen, Kazuhiro R Nitta, Pasi Rastas, Ekaterina Morgunova, Martin Enge, Mikko Taipale, Gonghong Wei, et al. Dna-binding specificities of human transcription factors. *Cell*, 152(1-2):327–339, 2013.
9. Jennifer M Binning, Tianjiao Wang, Priya Luthra, Reed S Shabman, Dominika M Borek, Gai Liu, Wei Xu, Daisy W Leung, Christopher F Basler, and Gaya K Amarasinghe. Development of rna aptamers targeting ebola virus vp35. *Biochemistry*, 52(47):8406–8419, 2013.
10. Brian R Baker, Rebecca Y Lai, McCall S Wood, Elaine H Doctor, Alan J Heeger, and Kevin W Plaxco. An electronic, aptamer-based small-molecule sensor for the rapid, label-free detection of cocaine in adulterated samples and biological fluids. *Journal of the American Chemical Society*, 128(10):3138–3139, 2006.
11. Mahmoud Labib, Anna S Zamay, Olga S Kolovskaya, Irina T Reshetneva, Galina S Zamay, Richard J Kibbee, Syed A Sattar, Tatiana N Zamay, and Maxim V Berezovski. Aptamer-based viability impedimetric sensor for bacteria. *Analytical chemistry*, 84(21):8966–8969, 2012.
12. Craig Tuerk and Larry Gold. Systematic evolution of ligands by exponential enrichment: Rna ligands to bacteriophage t4 dna polymerase. *science*, 249(4968):505–510, 1990.
13. Andrew D Ellington and Jack W Szostak. In vitro selection of rna molecules that bind specific ligands. *nature*, 346(6287):818–822, 1990.
14. Yue Zhao, David Granas, and Gary D Stormo. Inferring binding energies from selected binding sites. *PLoS Comput Biol*, 5(12):e1000590, 2009.
15. Arttu Jolma, Teemu Kivioja, Jarkko Toivonen, Lu Cheng, Gonghong Wei, Martin Enge, Mikko Taipale, Juan M Vaquerizas, Jian Yan, Mikko J Sillanpää, et al. Multiplexed massively parallel selex for characterization of human transcription factor binding specificities. *Genome research*, 20(6):861–873, 2010.
16. Gillian V Kupakuwana, James E Crill II, Mark P McPike, and Philip N Borer. Acyclic identification of aptamers for human alpha-thrombin using over-represented libraries and deep sequencing. *PloS one*, 6(5):e19395, 2011.
17. Peng Jiang, Susanne Meyer, Zhonggang Hou, Nicholas E Propson, H Tom Soh, James A Thomson, and Ron Stewart. Mpbind: a meta-motif-based statistical framework and pipeline to predict binding potential of selex-derived aptamers. *Bioinformatics*, 30(18):2665–2667, 2014.
18. Jimmy Caroli, Cristian Taccioli, A De La Fuente, Paolo Serafini, and Silvio Bicciato. Aptani: a computational tool to select aptamers through sequence-structure motif analysis of ht-selex data. *Bioinformatics*, 32(2):161–164, 2016.
19. Jimmy Caroli, Mattia Forcato, and Silvio Bicciato. Aptani2: update of aptamer selection through sequence-structure analysis. *Bioinformatics*, 36(7):2266–2268, 2020.
20. Ryoga Ishida, Tatsuo Adachi, Aya Yokota, Hidehito Yoshihara, Kazuteru Aoki, Yoshikazu Nakamura, and Michiaki Hamada. Raptranker: in silico rna aptamer selection from ht-selex experiment based on local sequence and structure information. *Nucleic acids research*, 48(14):e82–e82, 2020.

21. Namhee Kim, Joseph A Izzo, Shereef Elmetwaly, Hin Hark Gan, and Tamar Schlick. Computational generation and screening of rna motifs in large nucleotide sequence pools. *Nucleic acids research*, 38(13):e139–e139, 2010.
22. Jan Hoinka, Alexey Berezhnoy, Phuong Dao, Zuben E Sauna, Eli Gilboa, and Teresa M Przytycka. Large scale analysis of the mutational landscape in ht-selex improves aptamer discovery. *Nucleic acids research*, 43(12):5699–5707, 2015.
23. Qingtong Zhou, Xiaole Xia, Zhaofeng Luo, Haojun Liang, and Eugene Shakhnovich. Searching the sequence space for potent aptamers using selex in silico. *Journal of Chemical Theory and Computation*, 11(12):5939–5946, 2015.
24. Michael Hiller, Rainer Pudimat, Anke Busch, and Rolf Backofen. Using rna secondary structures to guide sequence motif finding towards single-stranded regions. *Nucleic acids research*, 34(17):e117–e117, 2006.
25. Phuong Dao, Jan Hoinka, Mayumi Takahashi, Jiehua Zhou, Michelle Ho, Yijie Wang, Fabrizio Costa, John J Rossi, Rolf Backofen, John Burnett, et al. Aptatrace elucidates rna sequence-structure motifs from selection trends in ht-selex experiments. *Cell systems*, 3(1):62–70, 2016.
26. Jan Hoinka, Elena Zotenko, Adam Friedman, Zuben E Sauna, and Teresa M Przytycka. Identification of sequence–structure rna binding motifs for selex-derived aptamers. *Bioinformatics*, 28(12):i215–i223, 2012.
27. Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831–838, 2015.
28. Hamid Reza Hassanzadeh and May D Wang. Deeperbind: Enhancing prediction of sequence specificities of dna binding proteins. In *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 178–183. IEEE, 2016.
29. Xiaoyong Pan, Peter Rijnbeek, Junchi Yan, and Hong-Bin Shen. Prediction of rna-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks. *BMC genomics*, 19(1):511, 2018.
30. Geoffrey E Hinton. Deep belief networks. *Scholarpedia*, 4(5):5947, 2009.
31. Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
32. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.
33. Jinho Im, Byungkyu Park, and Kyungsook Han. A generative model for constructing nucleic acid sequences binding to a protein. *BMC genomics*, 20(13):1–13, 2019.
34. Nathan Killoran, Leo J Lee, Andrew Delong, David Duvenaud, and Brendan J Frey. Generating and designing dna with deep generative models. *arXiv preprint arXiv:1712.06148*, 2017.
35. Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*, 2017.
36. Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.
37. Anders Krogh, Michael Brown, I Saira Mian, Kimmen Sjolander, and David Haussler. Hidden markov models in computational biology. applications to protein modeling. *Journal of molecular biology*, 235(5):1501–1531, 1994.
38. Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.
39. Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
40. Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. Batch bayesian optimization via local penalization. In *Artificial intelligence and statistics*, pages 648–657, 2016.
41. David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. Kriging is well-suited to parallelize optimization. In *Computational intelligence in expensive optimization problems*, pages 131–162. Springer, 2010.
42. The GPyOpt authors. GPyOpt: A bayesian optimization framework in python. `http://github.com/SheffieldML/GPyOpt`, 2016.
43. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
44. Catherine Lozupone, Shankar Changayil, Irene Majerfeld, and Michael Yarus. Selection of the simplest rna that binds isoleucine. *Rna*, 9(11):1315–1322, 2003.

45. Javier Gonzalez, Joseph Longworth, David C James, and Neil D Lawrence. Bayesian optimization for synthetic gene design. *arXiv preprint arXiv:1505.01627*, 2015.

46. Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

47. Hirotugu Akaike. Information theory as an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory. Akademiai Kiado, Budapest*, pages 276–281. B.N. Petrov, F. Csaki (Eds.), 1973.

48. Yasubumi Sakakibara, Michael Brown, Richard Hughey, I Saira Mian, Kimmen Sjölander, Rebecca C Underwood, and David Haussler. Stochastic context-free grammers for trna modeling. *Nucleic acids research*, 22(23):5112–5120, 1994.

49. Thomas D Schneider and R Michael Stephens. Sequence logos: a new way to display consensus sequences. *Nucleic acids research*, 18(20):6097–6100, 1990.

50. Gavin E Crooks, Gary Hon, John-Marc Chandonia, and Steven E Brenner. Weblogo: a sequence logo generator. *Genome research*, 14(6):1188–1190, 2004.

51. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

52. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

53. Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3, 2013.

54. Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

55. Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

56. Matthew T Weirauch, Ally Yang, Mihai Albu, Atina G Cote, Alejandro Montenegro-Montero, Philipp Drewe, Hamed S Najafabadi, Samuel A Lambert, Ishminder Mann, Kate Cook, et al. Determination and inference of eukaryotic transcription factor sequence specificity. *Cell*, 158(6):1431–1443, 2014.