End-to-end RNA Secondary Structure Prediction using Autoregressive Deep Convolutional Neural Network

1 Introduction

Unlike DNA which typically forms stable double helix between two molecules, RNA is highly flexible so a single RNA molecule can fold onto itself by forming base pairs via hydrogen bond, including Watson-Crick pairs A-U, G-C and non-canonical pairs such as G-U. Base pairs form local structures like stems and loops, which assemble into the global secondary structure.

State-of-the-art RNA secondary structure prediction algorithms like ViennaRNA[1] and Mfold[2] are based on thermodynamics. Free energy of each type of local structure is measured experimentally, and total free energy is assumed to be the sum of local energy terms, which is minimized efficiently via dynamic programming (DP). Although researchers have worked out a large set of local structure types and their associated formulation for free energy calculation, there is no guarantee that it is accurate and complete. There have been efforts to fine-tune the free energy parameters by training on experimentally solved structures[3], but it is still limited by the known set of local structure types. Another limitation of DP-based approaches is that it is incapable of predicting nested base pairs that appear in structures such as pseudoknot.

On the other hand, over the last decade, a combination of RNA structure probing and high-throughput sequencing has enabled the measurement of genome-wide RNA structures at single nucleotide resolution in multiple organisms and cell types[4, 5]. Due to the level of noise and missing data present in high-throughput experiments, a more flexible modeling approach is needed which can be trained on different types of data.

In this work, we propose a autoregressive deep convolutional neural network that can be trained end-to-end on sequences and structures. Given an input RNA sequence, the model can generate a distribution of structures, including ones with pseudoknot.

1.1 Problem formulation

Given an RNA sequence of length L, we are interested in all possible secondary structures. To represent a specific secondary structure, there are three commonly used conventions: (1) undirected graph, where each node is a base in the sequence, and each edge represents base pairing. (2) upper triangular matrix (excluding the diagonal) of size $L \times L$ with binary values, where a value of 1 at (i, j) represents base pairing between sequence position i and j, and 0 represents no base paring. (3) dot-bracket notation of length L where unpaired bases are denoted as dots, and paired bases are represented as left and right brackets.

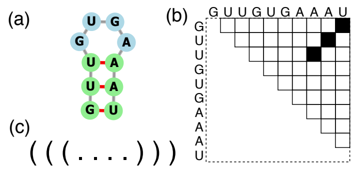

Figure 1: Three different ways to represent secondary structure. (a) undirected graph, generated by [6], (b) upper triangular binary matrix, (c) dot-bracket notation.

As an example, for a short RNA sequence GUUGUGAAAU, one possible structure it can take on consists of a stem and a loop, as seen in Fig 1(a), represented by an undirected graph. Such structure can also be represented by a 10×10 upper triangular matrix with all 0's, except for positions $(1, 10)$, $(2, 9)$ and $(3, 8)$, all being 1, as shown in Fig 1(b). This contiguous stretch of 1's along the diagonal corresponds to the stem formed by the three base pairs: G-U, U-A and U-A. The equivalent dot-bracket notation is shown in Fig 1(c), where the stem is represented by three pairs of left-right brackets.

1.2 Related work

Several groups have tried to apply neural network in modeling RNA secondary structure. To the best of our knowledge, existing work either solves a partial problem, or attempts at modeling the problem end-to-end, but requires complicated post-processing to yield valid structure prediction.

Wu[7] presented a convolution neural network to predict the free energy of local structure motifs. Although it shows promising results on modeling the free energy of short structure motif from experimental data and on estimating the free energy for a *given* structure, it does not solve the problem of predicting structure from sequence.

Researchers have also framed the problem as a sequence-to-sequence learning task. The proposed models either predict the pairing probability for each base, or some forms of the dot-bracket notation. Willmott et. al[8] proposed a bidirectional LSTM that predicts the per-position probability for 3 classes: paired, unpaired, and end-of-sequence. The model was trained on sequences that belong to the same RNA family, so it is unclear whether the model generalizes to other types of RNAs. Zhang et. al[9] proposed a convolutional neural network to predict the dot-bracket notation, where each position is encoded as a 3-class softmax: left bracket, right bracket and dot. In order for a dot-bracket notation to yield valid structure, it has to follow a few syntactic constraints. For example, each left bracket needs to have a corresponding right bracket. As the authors mentioned in the paper, output produced by the neural network was not guaranteed to be valid, thus needed to be further processed by dynamic programming to yield a valid structure prediction. Wang et. al[10] used bidirectional LSTM to predict the dot-bracket notation as a 7-class softmax, which includes additional bracket notations to allow for pseudoknot. Similar to [9], the prediction did not yield a valid structure, and needed further post-processing. Furthermore, they used a dataset of only four RNA families, but reported performance based on randomized training and validation set split. Since there is high level of sequence similarity within the same family, generalization performance of the model remains unclear.

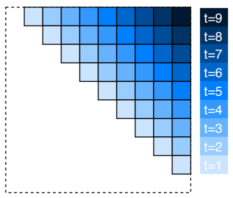

Figure 2: Generative process implied by Equation 1.

2 Method

In this work, we propose a autoregressive deep convolutional neural network that is trained end-to-end on dataset with sequences and structures. The model is capable of generating a distribution of structures conditioned on the input sequence, including structures with pseudoknot.

We formulate the predictive task as a conditional generative process. Specifically, given an input sequence of length L: $\boldsymbol{x} = (x_1, x_2, \dots x_L)$, we want to predict a distribution of structures conditioned on the sequence $P(\boldsymbol{Y} \mid \boldsymbol{x})$, where the structure $\boldsymbol{Y}$ is represented by an upper triangular matrix of size $L \times L$, as defined in Fig 1(b).

We factorize the conditional distribution as follows:

$$\begin{aligned} P(\boldsymbol{Y} \mid \boldsymbol{x}) =& P(\{y_{i,i+1}\}_{i=1,\dots L-1} \mid \boldsymbol{x}) \\ & P(\{y_{i,i+2}\}_{i=1,\dots L-2} \mid \boldsymbol{x}, \{y_{i,i+1}\}_{i=1,\dots L-1}) \\ & \dots \\ & P(y_{1,L} \mid \boldsymbol{x}, \{y_{i,j}\}^{i \neq 1}_{j \neq L}) \end{aligned} \tag{1}$$

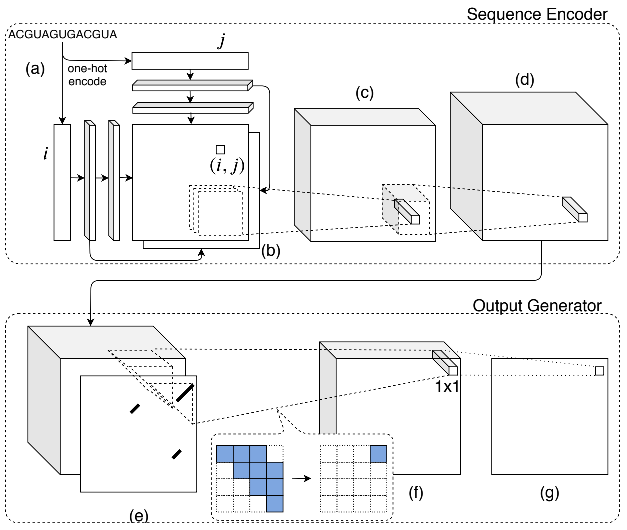

Figure 3: Proposed model architecture. Top: sequence encoder. Bottom: output generator.

The generative process implied by such formulation is illustrated in Fig 2. Starting from the slice adjacent to the off-diagonal line (light blue with $t = 1$), we generate one off-diagonal slice at a time, conditioned on the input sequence and the previous slices. This process continues until we fill the upper triangular matrix, where the last entry (dark blue, with $t = 9$) is conditioned on the input and the entire upper triangular matrix except for itself. Intuitively, the distance to the off-diagonal line is analogous to the "timestamp" as in classic autoregressive models.

2.1 Architecture

We propose an architecture that generates global structure from interactions between different positions in the sequence, with neither assumptions on the types of local structures nor hard-coded parameters, such that the entire model can be trained end-to-end using only sequences and structures. The architecture consists of the following components:

1. Two sets of 1D convolutional (conv) layers on the one-hot-encoded sequence are run in parallel, with each set having multiple layers at different resolutions, as shown in Fig 3 (a) (showing only two layers for simplicity). Activations of the 1D conv layer, one from each set, are used to form a 2D feature map via inner product along the feature dimension, as shown in Fig 3 (b). This is inspired by the fact that base pairing is not only affected by the two bases

but also surrounding sequences. Such architecture enables the neural network to learn features that affect interaction between any two positions in the sequence.

2. Multiple 2D feature maps, each produced by one pair of 1D activations at a specific layer, are concatenated, as shown in Fig 3 from (b) to (c). This is followed by several 2D conv layers, as shown in Fig 3 from (c) to (d) (showing only one layer for simplicity).
3. Activations of the last 2D conv layer is concatenated with output from the previous "timestamp" y^{t-1}, and processed by multiple layers of masked 2D conv layers, as shown in Fig 3 from (e) to (f) (showing only one layer for simplicity). Masking ensures that the output node only has access to information in the lower left corner ("past" *w.r.t.* the output node) of the matrix, which enables the model to generate output in autoregressive fashion once unrolled in inference mode (see Section 2.3 for mode details).
4. Finally, a fully connected layer along the feature dimension with sigmoid activation produces an output between 0 and 1, for each position in the upper triangular matrix, as illustrated in Fig 3 from (e) to (f).

As shown in Fig 3, we refer to 1-2 as the sequence encoder, and 3-4 as the output generator.

2.2 Training

We constructed a synthetic dataset for training the model, by sampling 50000 random sequences. For each sequence, the length is sampled uniformly from 10 to 100, and each base is sampled *i.i.d.* uniformly from $\{A, C, G, U\}$. For each sequence, we ran RNAfold[1] with default parameters and recorded the minimum free energy structure.

The model has 5 pairs of 1D-conv layers, each having 256 filters, with filter width and dilation rate being $(7, 1), (5, 2), (5, 2), (5, 4), (5, 4)$. There are 5 masked 2D-conv layers, each having 50 filters, with filter size and dilation rate being $([3, 3], 1), ([3, 3], 2), ([5, 5], 2), ([9, 9], 4), ([9, 9], 4)$. Fully connected layer has 20 hidden units. All layers use ReLU activation, except for the output layer being sigmoid. 40000 sequences were used for training and 10000 for validation, with a batch size of 10 sequences. Adam optimizer[11] was used with learning rate 0.001 and the model was trained over 200 epochs with early stopping.

For each batch, we zero-pad the sequence array and structure matrix to the maximum length in the batch. When computing the loss and gradient, both the lower triangular and the padding entries are masked. At training time, prediction, loss, and gradient of all positions in the output are computed in parallel, by passing in the target structure matrix as both the input and target.

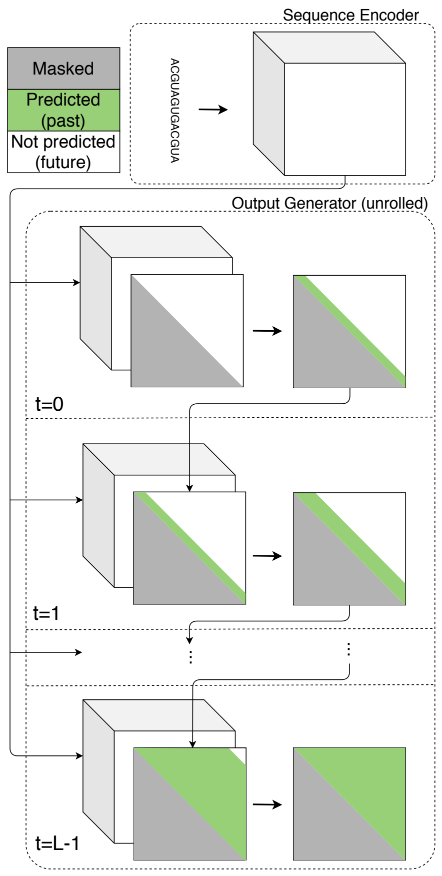

Figure 4: Sample structures conditioned on the input sequence.

2.3 Inference

At test time, we sample structures conditioned on the input sequence. As shown in Fig 4, we first pass the sequence through the sequence encoder network to get the 2D feature map. We then initialize the structure matrix with all zeros, concatenate it with the 2D feature map, and sample one slice at a time until the upper triangular matrix is filled with sampled values. At each step, we sample a binary label for each position in the current slice based on the Bernoulli probability predicted by the model. To ensure the sampled structure is valid, when sampling the label for location (i, j), if the i-th or j-th position is already paired with another position (from samples in previous timestamps), we set $y_{i,j}$ to 0 without sampling from the model output. For a sequence of length L, we run $L - 1$ steps sequentially.

3 Result

3.1 Test set performance

To evaluate the model on unseen data, we generated a test set with 100 sequence of length between 10 and 100, using the same procedure that generated the training set. For each sequence in the test set, we ran RNAfold and sampled 100 structures from the ensemble. We also used our model to sample 100 structures from the output distribution conditioned on each sequence. To avoid evaluating on low probability samples, structures occurring only once were discarded. To compare the two structure distributions, for each unique structure produced by RNAfold, we computed sensitivity (number of correctly predicted base pairs divided by number of true base pairs) and positive predictive value (PPV, a.k.a. precision, number of correctly predicted base pairs divided by number of predicted base pairs) against all unique structures generated by our model and recorded the best one, where best is defined by largest sensitivity + PPV. This indicates how well the model recovers each of the structures produced by RNAfold. Fig 5 shows a the distribution of sensitivity and PPV across all structures in all sequences. It can be observed that majority of the RNAfold-generated structures can be recovered with larger than 0.8 sensitivity and PPV.

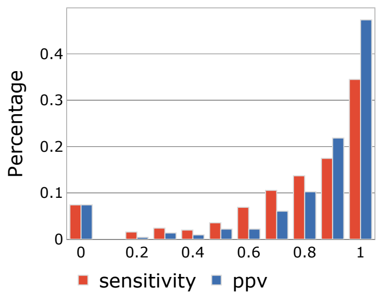

Figure 5: Performance on test set.

3.2 Structures with pesudoknot

In addition to generating structures similar to the state-of-the-art models, our model is also capable of predicting structures it has never seen during training, since it does not incorporate hard-wired rules on how local structures assemble into global structure. As an example, we use the sequence of mouse mammary tumor virus (MMTV), whose secondary structure contains pseudoknot as measured by nuclear magnetic resonance (NMR), as shown in Fig 6(a). Minimum free energy structure predicted by RNAfold takes a quite different form, as shown in Fig 6(b). In contrast, when we sampled 100 structures from our model, we observed a diverse set of structures, including the one predicted by RNAfold, as shown in Fig 6(c3), and more interestingly, the pseudoknot structure from NMR, as shown in Fig 6(c1).

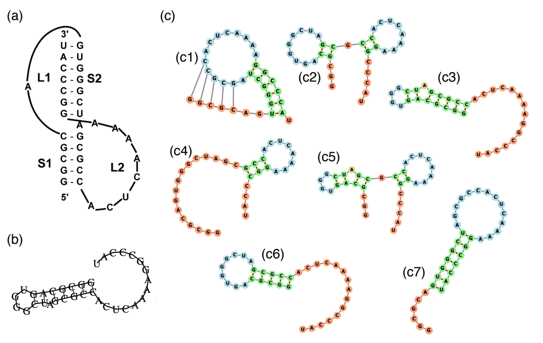

Figure 6: Secondary structure of mouse mammary tumor virus (MMTV): (a) measured by nuclear magnetic resonance (NMR)[12], (b) predicted by RNAfold, (c) generated by our model (only showing unique structures).

3.3 Sequence design via gradient ascent

One benefit of differentiable model is that we can answer interesting questions like: given my current input sequence, what (small) changes can be made to maximize the pairing probability of two positions. We illustrate this using a short RNA sequence GAUCACCUUUGGAUC. The sequence is short enough such that vast majority of the structures in the distribution are identical, as shown in Fig 7(a). We start with the current predicted structure and want to maximize base pairing at position (i, j). In order to find small changes to be made on this sequence, we computed the gradient of the output node at (i, j) with respect to all the input nodes $\partial y_{i,j}/\partial \boldsymbol{x}$. We ran 100 gradient ascent iterations with step size 0.01. In each step, after adding the gradient (times step size) onto the input, we re-normalize the input so values along the feature dimension sum up to 1. After all 100 iterations, we convert the input matrix back to a sequence by taking the nucleotide with max score for each position. Fig 7(b) and (c) shows the resulting sequence and structure for $i = 5, j = 11$ and $i = 6, j = 10$, respectively.

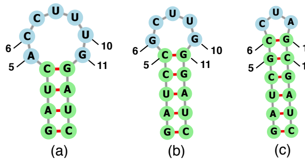

Figure 7: Sequence design via gradient ascent to maximize base pairing. (a) original structure, (b) maximize pairing at position (5, 11), (c) maximize pairing at position (6, 10).

References

[1] Ronny Lorenz, Stephan H Bernhart, Christian Höner Zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. Viennarna package 2.0. *Algorithms for molecular biology*, 6(1):26, 2011.

[2] Michael Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic acids research*, 31(13):3406–3415, 2003.

[3] Mirela Andronescu, Anne Condon, Holger H Hoos, David H Mathews, and Kevin P Murphy. Efficient parameter estimation for rna secondary structure prediction. *Bioinformatics*, 23(13):i19–i28, 2007.

[4] Stefanie A Mortimer, Mary Anne Kidwell, and Jennifer A Doudna. Insights into rna structure and function from genome-wide studies. *Nature reviews Genetics*, 15(7):469, 2014.

[5] Philip C Bevilacqua, Laura E Ritchey, Zhao Su, and Sarah M Assmann. Genome-wide analysis of rna secondary structure. *Annual review of genetics*, 50:235–266, 2016.

[6] Peter Kerpedjiev, Stefan Hammer, and Ivo L Hofacker. Forna (force-directed rna): simple and effective online rna secondary structure diagrams. *Bioinformatics*, 31(20):3377–3379, 2015.

[7] Michelle J Wu. Convolutional models of rna energetics. *bioRxiv*, page 470740, 2018.

[8] Devin Willmott, David Murrugarra, and Qiang Ye. State inference of rna secondary structures with deep recurrent neural networks.

[9] Hao Zhang, Chunhe Zhang, Zhi Li, Cong Li, Xu Wei, Borui Zhang, and Yuanning Liu. A new method of rna secondary structure prediction based on convolutional neural network and dynamic programming. *Frontiers in genetics*, 10, 2019.

[10] Linyu Wang, Yuanning Liu, Xiaodan Zhong, Haiming Liu, Chao Lu, Cong Li, and Hao Zhang. Dmfold: A novel method to predict rna secondary structure with pseudoknots based on deep learning and improved base pair maximization principle. *Frontiers in genetics*, 10:143, 2019.

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[12] David W Staple and Samuel E Butcher. Pseudoknots: Rna structures with diverse functions. *PLoS biology*, 3(6):e213, 2005.