RNA Secondary Structure Prediction By Learning Unrolled Algorithms

RNA secondary structure prediction is one of the oldest problems in bioinformatics, which has been studied for more than 40 years. Given the input RNA sequence, we want to obtain the pairing information for each base in the sequence. We know that only specific pairings are allowed ({A,U}, {C,G}, {U,G}), while the other pairs are not physically stable and thus not allowed. Traditionally, people would assign a particular energy value to each pair and figure out the specific secondary structure pattern with the lowest summarized energy of all the pairs. With this problem formulation, researchers usually utilize dynamic programming (DP) to resolve it. However, as we know, DP can be slow. Also, the DP-based methods have reached the prediction performance bottleneck, with the F1 score being around 0.6. Furthermore, the canonical DP-based methods are unable to handle pseudoknot prediction, where there is a jump in the paring (Figure 1).

To resolve the limitations of the DP-based methods, we address it from an entirely new angle. Instead of considering it as an optimization problem, we regard the task as a translation with constraints problem. Given the sequence, we translate it into a binary contact map matrix, which has a one-to-one mapping against a certain secondary structure (Figure 1). When resolving the translation problem, we also need to consider the constraints for the original problem: 1) only specific pairs are allowed; 2) no sharp loops are allowed; 3) it is a matching problem. We should enforce those constraints into the outputted matrix to ensure the corresponding secondary structure satisfying the physical properties of the problem. Notice that such a formulation can cover the pseudoknot prediction naturally.

The overall idea of E2Efold for solving the above problem is shown in Figure 1. Firstly, we resolve a translation without constraints problem, whose output may not satisfy all the constraints. Then, we further utilize constrained optimization to enforce the constraints and adjust the result from the first step, obtaining the final solution. However, the above two-step solution is not our ultimate solution. Eventually, we combine those two steps into one integrated deep learning model, proposing a new deep learning architecture, which has the constrained optimization embedded in the network. Given the RNA sequence, the model can output a contact map in one step, which satisfies the constraints directly, without intermediate output.

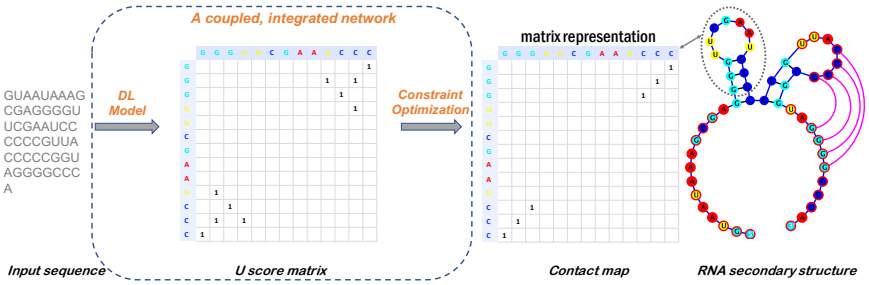

Figure 1: The overall idea of E2Efold.

Regarding the translation without constraints problem, inspired by Transformer, we propose a deep learning architecture (Figure 2 left part, before PP Network) to translate the input sequence into a U score matrix. The Transformer encoder layers can aggregate information from the original sequence and the position embeddings, outputting the element-wise embedding for each base. Then, considering all the pairs within the sequence, we convert the element-wise embeddings into the pair-wise embeddings, which are further processed by 2D convolutional layers to produce the U score matrix. As for the constrained optimization, which helps us enforce the problem-specific constraints, we optimize against the correlation between the final binary contact map and the U matrix, with the final contact map satisfying all the constraints. After translating the aforementioned constraints into the mathematical form and converting the constrained optimization into the equivalent unconstrained form, we can use the gradient descent algorithm to resolve the problem efficiently and produce the final contact map. Essentially, it is the gradient descent algorithm, which is the solution for the constrained optimization problem, that helps us enforce the constraints.

To integrate the above two steps into one model, we unroll the gradient descent algorithm to a certain T steps, hoping that the unrolled algorithm can enforce the same constraints as the original algorithm. Then, the unrolled algorithm becomes the building block of the integrated network (PP network in Figure 2). More specifically, each iteration in the unrolled algorithm becomes one layer in the PP network, and the operations within the network are the same as the unrolled algorithm. Thus, the total number of layers in the subnetwork is T. Regarding the parameters in the network, all the hyper-parameters in the original gradient descent algorithm, such as learning rate, which are usually handcrafted, become the learnable parameters. Essentially, we design such a subnetwork with the same architecture as the unrolled gradient descent algorithm to embed the constraints contained in the original algorithm into the network.

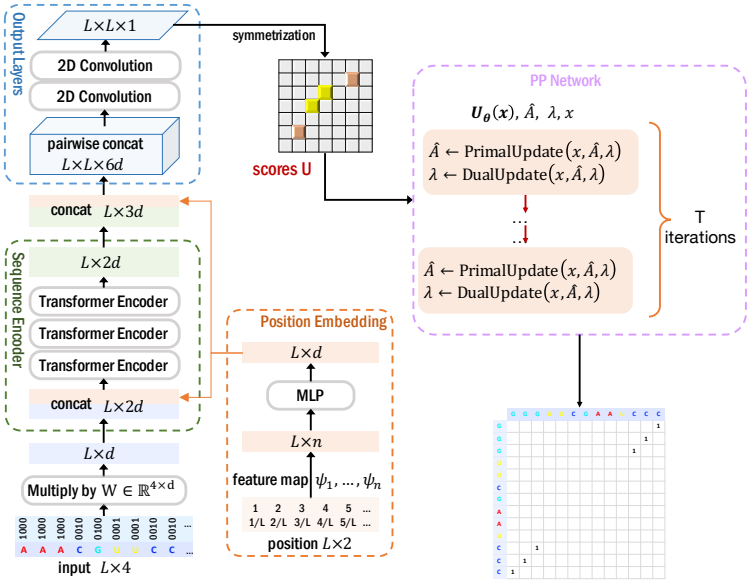

Figure 2: The E2Efold architecture.

After designing the PP network with constraints embedded, we couple it with the U score network (Figure 2). During training, we train the two networks together, differentiating loss function through the PP network towards the upstream network. By doing so, the U network can know the existence of the PP network as well as the constraints embedded. After being trained, given the RNA sequence, the model is very likely to output a contact map, satisfying the constraints directly.

With comprehensive experiments, we demonstrate the superior performance of E2Efold. It predicts significantly better structures compared to the previous state-of-the-art methods, with the testing F1 score being 0.82 (Table 1), while being as efficient as the fastest algorithm in terms of inference time (Table 2). Regarding pseudoknot prediction, E2Efold can outperform the previous methods by 25% (Table 5 in the full paper). Furthermore, we randomly select a 16S rRNA, visualizing the predictions in Figure 3. It suggests that E2Efold can indeed predict a much better RNA secondary structure, compared to the other methods, which is very similar to the ground truth. The original paper has been published as an oral paper in ICLR 2020, whose acceptance rate is 48 out of 2559. The code of E2Efold is available at https://github.com/ml4bio/e2efold.

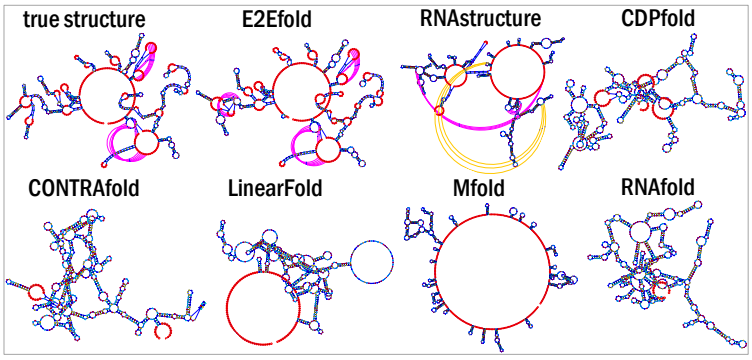

Figure 3: Visualization of a 16S rRNA, DQ170870.

Table 1: Results on RNAStralign test set.

Method	Prec	Rec	F1	Prec(S)	Rec(S)	F1(S)
E2Efold	**0.866**	**0.788**	**0.821**	**0.880**	**0.798**	**0.833**
CDPfold	0.633	0.597	0.614	0.720	0.677	0.697
LinearFold	0.620	0.606	0.609	0.635	0.622	0.624
Mfold	0.450	0.398	0.420	0.463	0.409	0.433
RNAstructure	0.537	0.568	0.550	0.559	0.592	0.573
RNAfold	0.516	0.568	0.540	0.533	0.587	0.558
CONTRAfold	0.608	0.663	0.633	0.624	0.681	0.650

Table 2: Inference time on RNAStralign.

Method		total run time	time per seq
E2Efold (Pytorch)		**19m (GPU)**	**0.40s**
CDPfold (Pytorch)		440m*32 threads	300.107s
LinearFold	(C)	20m	0.43s
Mfold	(C)	360m	7.65s
RNAstructure	(C)	3 days	142.02s
RNAfold	(C)	26m	0.55s
CONTRAfold	(C)	1 day	30.58s