RNA Secondary Structure Prediction By Learning Unrolled Algorithms

Xinshi Chen*[1], Yu Li*[2], Ramzan Umarov[2], Xin Gao[2], Le Song[1,3]

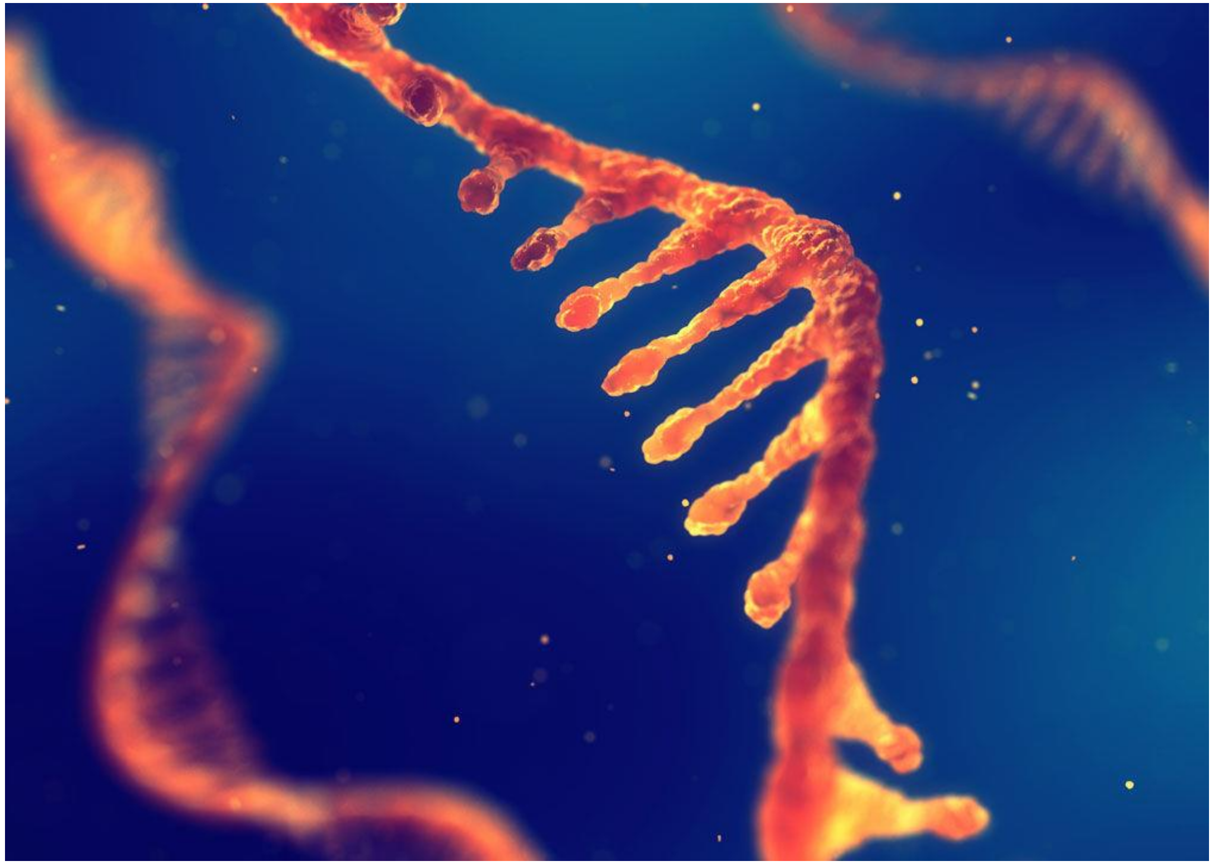
[1]Georgia Tech, [2]KAUST, [3]Ant Financial

ICLR 2020

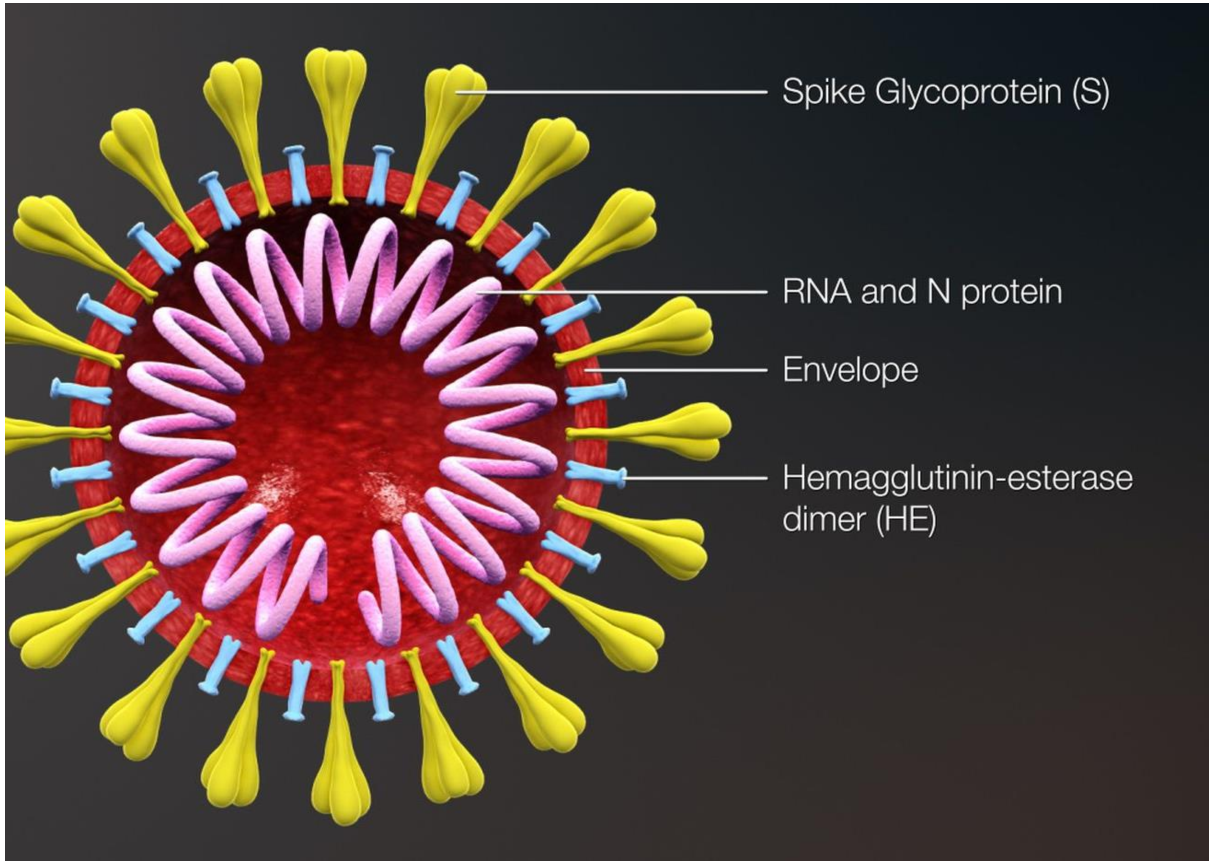
* Equal contribution

Ribonucleic Acid (RNA)

RNA (Ribonucleic acid)

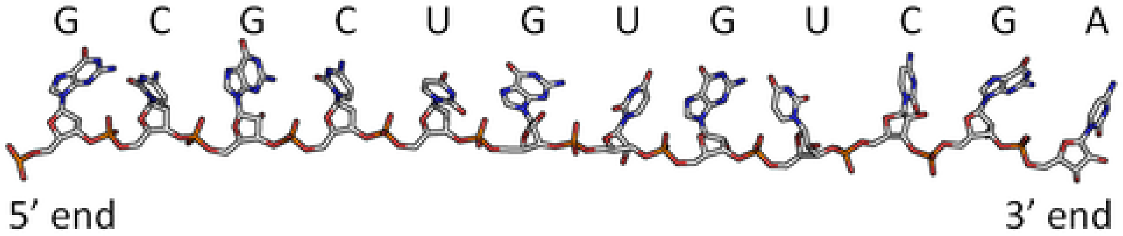


RNA Virus (e.g., COVID-19)



RNA Primary Structure

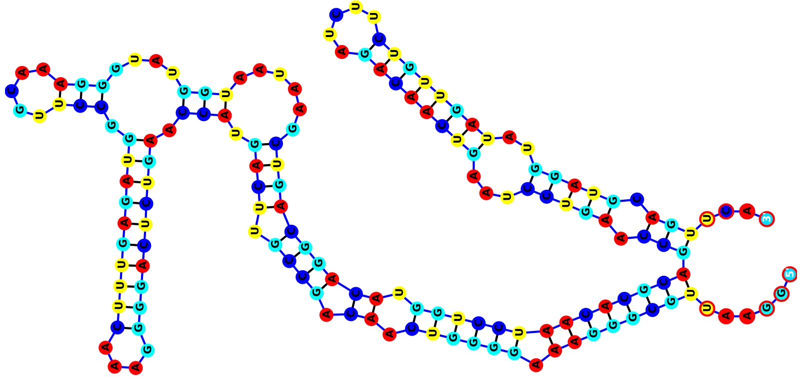
Primary Structure



$$\boldsymbol{x} = (x_1, x_2, \ldots, x_L), \qquad x_i \in \{\mathrm{A}, \mathrm{U}, \mathrm{C}, \mathrm{G}\}$$

RNA Secondary Structure

Secondary Structure



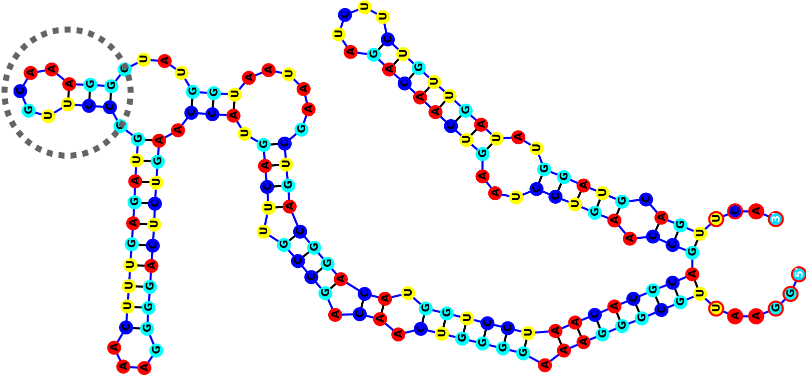
$$\boldsymbol{A}^* \in \{0,1\}^{L\times L}$$

$\boldsymbol{A}^*(i,j) = 1$ if the bases (x_i, x_j) are paired.

RNA Secondary Structure

Secondary Structure

	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

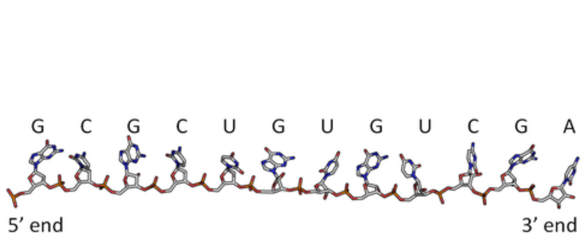


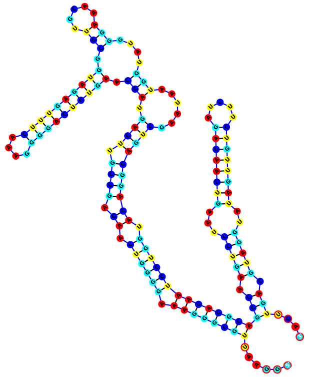
$$\boldsymbol{A}^* \in \{0,1\}^{L\times L}$$

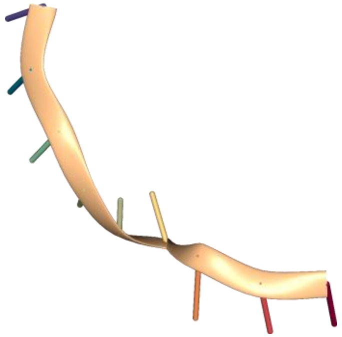
$\boldsymbol{A}^*(i,j) = 1$ if the bases (x_i, x_j) are paired.

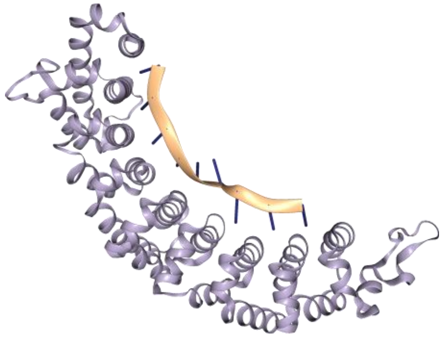
High Order Structures of RNA

Primary Structure → Secondary Structure → 3D Structure → Function





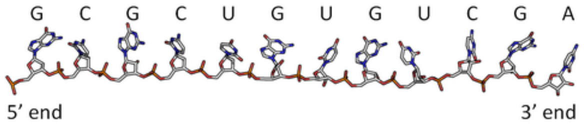


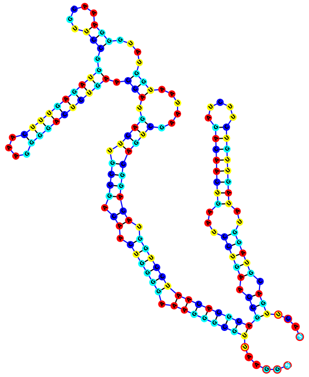


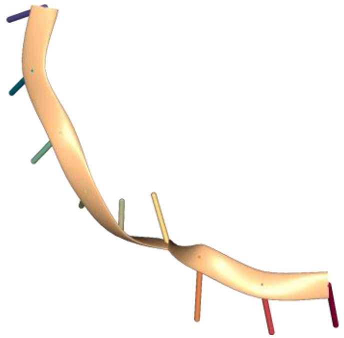
$\boldsymbol{x} = (x_1, x_2, \dots, x_L)$ $\boldsymbol{A}^* \in \{0,1\}^{L \times L}$

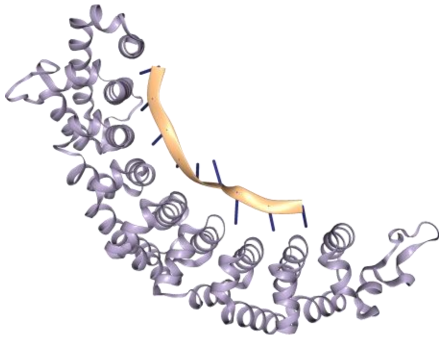
RNA Secondary Structure Prediction

Primary Structure → Secondary Structure → 3D Structure → Function









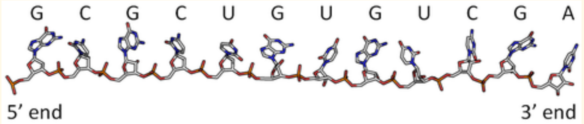
$\boldsymbol{x} = (x_1, x_2, \dots, x_L)$

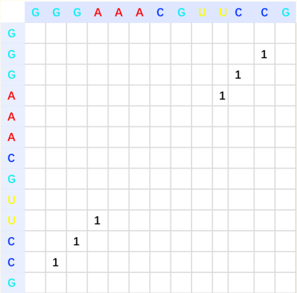
$\boldsymbol{A}^* \in \{0,1\}^{L \times L}$

Existing Method: Energy Minimization Based Model

$$\boldsymbol{A}^* = \underset{\boldsymbol{A}\in\{0,1\}^{L\times L}}{\operatorname{argmin}} \; E(\boldsymbol{x}, \boldsymbol{A})$$

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_L) \longrightarrow E(\boldsymbol{x}, \boldsymbol{A}) \xrightarrow[\boldsymbol{A}\,\in\,\{0,1\}^{L\times L}]{\text{energy minimization}} \boldsymbol{A}^*$$





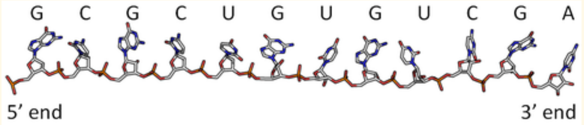
	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

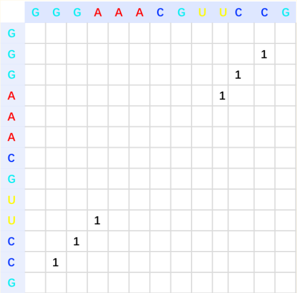
- ✗ $E(\boldsymbol{x}, \boldsymbol{A})$ can be inaccurate
- ✗ Intractable minimization (exponential in $\boldsymbol{L}$)

Existing Method: Energy Minimization Based Model

$$\boldsymbol{A}^* = \underset{\substack{\cancel{\boldsymbol{A}\in\{0,1\}^{L\times L}} \\ \boldsymbol{A}\,\in\,\textbf{Nested Structures}}}{\text{argmin}} E(\boldsymbol{x},\boldsymbol{A})$$

$$\boldsymbol{x} = (x_1, x_2, \dots, x_L) \longrightarrow E(\boldsymbol{x},\boldsymbol{A}) \xrightarrow[\substack{\cancel{\boldsymbol{A}\in\{0,1\}^{L\times L}} \\ \boldsymbol{A}\,\in\,\textbf{Nested Structures}}]{\text{energy minimization}} \boldsymbol{A}^*$$





	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

✗ $E(\boldsymbol{x},\boldsymbol{A})$ can be inaccurate

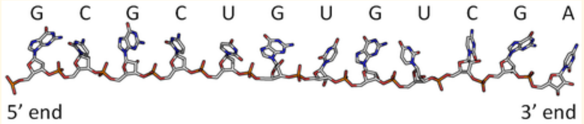
~~✗ Intractable minimization (exponential in L)~~

Assume $\boldsymbol{A}^*$ has a ***nested structure***

Existing Method: Energy Minimization Based Model

$$\boldsymbol{A}^* = \underset{\substack{\cancel{\boldsymbol{A}\in\{0,1\}^{L\times L}} \\ \boldsymbol{A} \in \textbf{Nested Structures}}}{\operatorname{argmin}} E(\boldsymbol{x},\boldsymbol{A})$$

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_L) \longrightarrow E(\boldsymbol{x},\boldsymbol{A}) \xrightarrow[\substack{\cancel{\boldsymbol{A}\in\{0,1\}^{L\times L}} \\ \boldsymbol{A} \in \textbf{Nested Structures}}]{\text{energy minimization}} \boldsymbol{A}^*$$



	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

✗ $E(\boldsymbol{x},\boldsymbol{A})$ can be inaccurate

~~✗ Intractable minimization (exponential in $\boldsymbol{L}$)~~

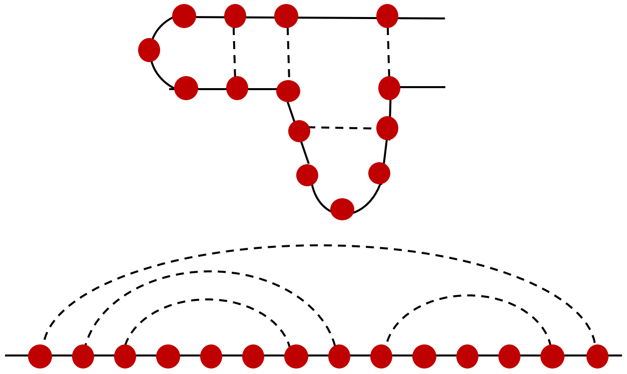
Assume $\boldsymbol{A}^*$ has a ***nested structure***

✓ Dynamic programming (DP)

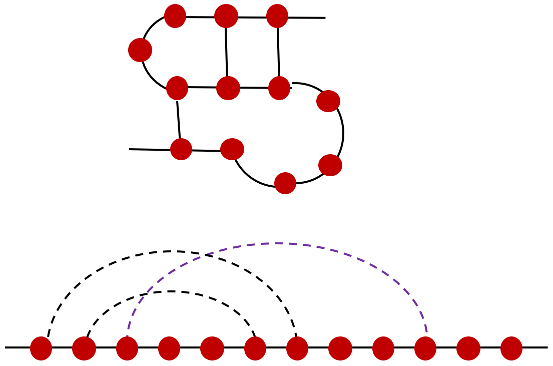
✓ Tractable minimization $\boldsymbol{O}(L^3)$

Existing Method: Energy Minimization Based Model

Nested Structure



Non-nested Structure (pseudoknot)



present in around **40%** of the RNAs

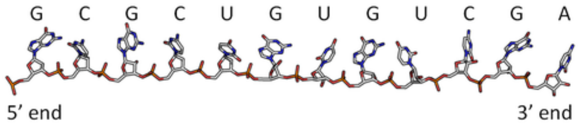
✗ Cannot handle more complicated structures (**pseudoknots**)

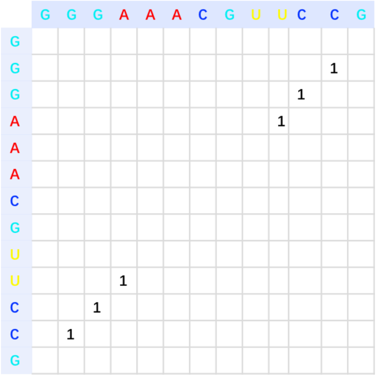
✗ $\boldsymbol{O}(L^3)$ is still slow

Existing Method: Direct Mapping

- **Deep Network $\boldsymbol{F_\theta}$**

$$\boldsymbol{x} = (x_1, x_2, \dots, x_L) \xrightarrow{F_\theta} \boldsymbol{A}^* \in \{0,1\}^{L\times L}$$





	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

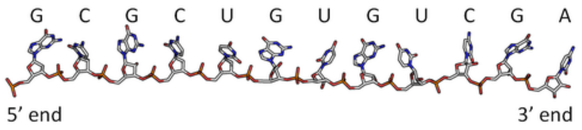
✓ Can predict both nested structures and pesudoknots

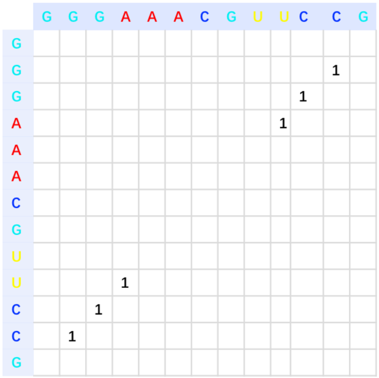
✓ Avoids the expensive minimization step

Existing Method: Direct Mapping

- **Deep Network $\boldsymbol{F_\theta}$**

$$\boldsymbol{x} = (x_1, x_2, \dots, x_L) \xrightarrow{F_\theta} \boldsymbol{A}^* \in \{0,1\}^{L\times L}$$





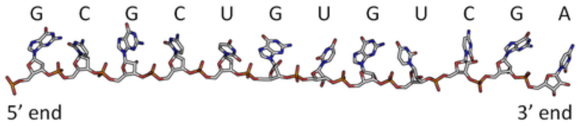
	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

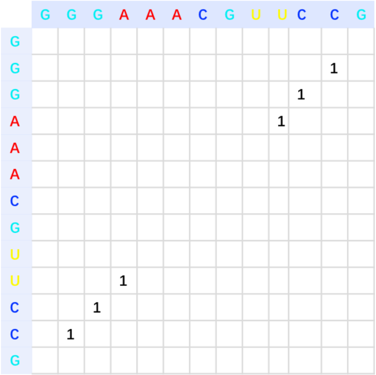
- **New Challenges**
  - RNA secondary structure $\boldsymbol{A}^*$ needs to obey some **hard constraints**.
    - ➢ Only {A - U, C - G, G - U} are valid pairings.
    - ➢ No sharp loops are allowed.
    - ➢ No overlap of pairs is allowed, i.e., it is a matching.

Existing Method: Direct Mapping

- **Deep Network $\boldsymbol{F_\theta}$**

$$\boldsymbol{x} = (x_1, x_2, \dots, x_L) \xrightarrow{F_\theta} \boldsymbol{A}^* \in \{0,1\}^{L \times L}$$





	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

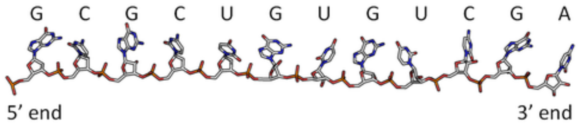
- **New Challenges**
  - RNA secondary structure $\boldsymbol{A}^*$ needs to obey some **hard constraints**.

    ★ How to make the output of F_θ satisfy the constraints?

Existing Method: Direct Mapping

- **Deep Network $\boldsymbol{F_\theta}$**

$$\boldsymbol{x} = (x_1, x_2, \dots, x_L) \xrightarrow{F_\theta} \boldsymbol{A}^* \in \{0,1\}^{L\times L}$$



	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

- **New Challenges**
  - RNA secondary structure $\boldsymbol{A}^*$ needs to obey some **hard constraints**.

    ★ How to make the output of F_θ satisfy the constraints?
  - The number of RNA **data** points is **limited**.

    ★ Difficult to learn the constraints directly from data.

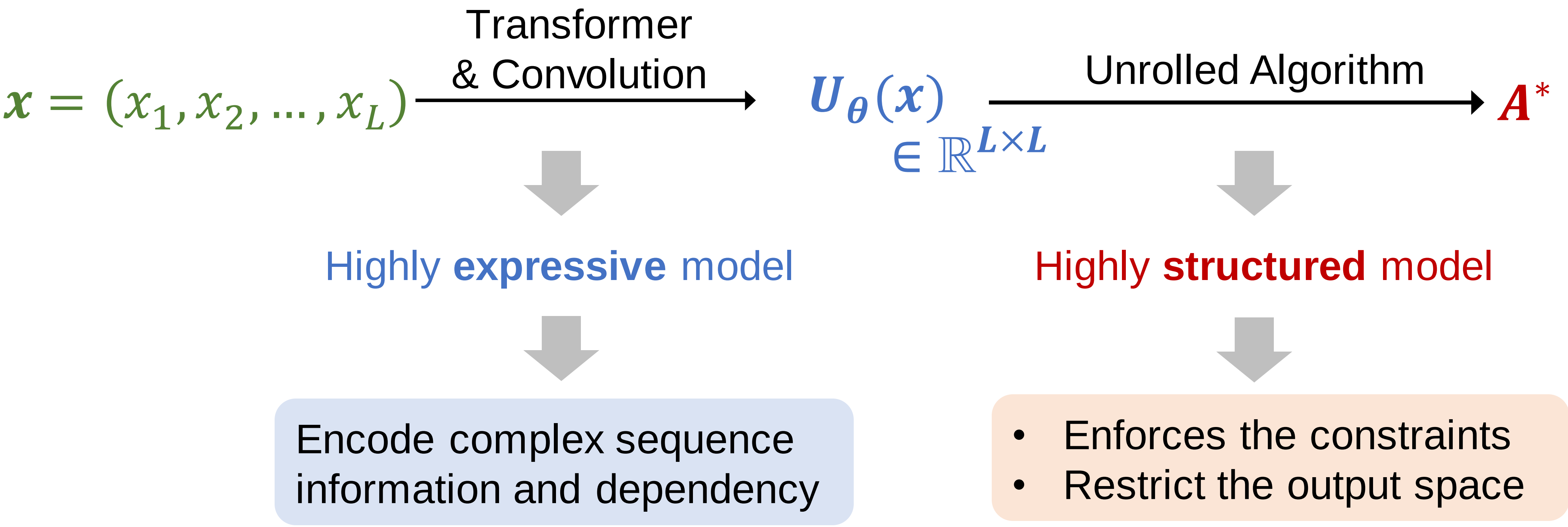
    ★ Overfitting issue

E2Efold: Enforce Constraints with Deep Architecture

$$\boldsymbol{x} = (x_1, x_2, \dots, x_L) \xrightarrow[]{\text{Transformer}\ \text{\& Convolution}} \boldsymbol{U}_{\boldsymbol{\theta}}(\boldsymbol{x}) \xrightarrow{\text{Unrolled Algorithm}} \boldsymbol{A}^*$$

	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

E2Efold: Enforce Constraints with Deep Architecture



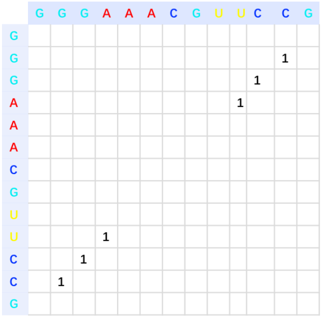
$$\boldsymbol{x} = (x_1, x_2, \dots, x_L) \xrightarrow{\text{Transformer \& Convolution}} \boldsymbol{U_\theta(x)} \in \mathbb{R}^{\boldsymbol{L \times L}} \xrightarrow{\text{Unrolled Algorithm}} \boldsymbol{A}^*$$

Highly **expressive** model

Encode complex sequence information and dependency

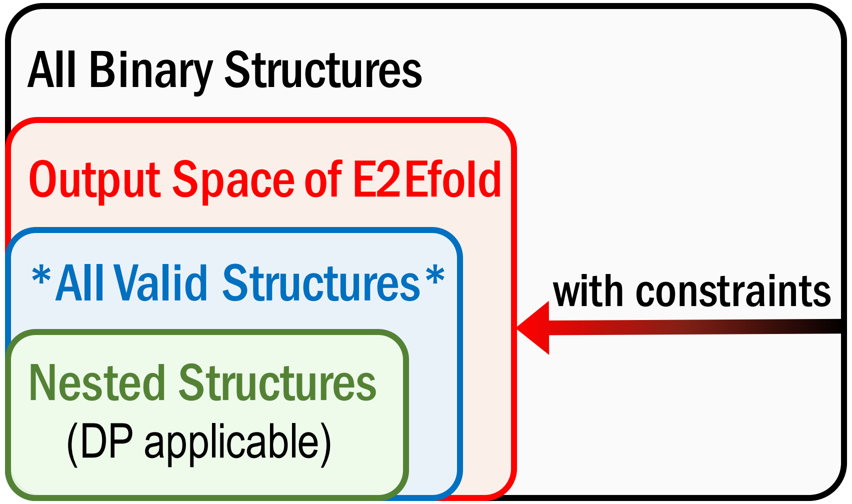
Highly **structured** model

- Enforces the constraints
- Restrict the output space



	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

Model Space Comparison



DP-based methods:

- $O(L^3)$ complexity
- Can not predict non-nested structures

A naïve neural network:

- Hard to enforce constraints
- Overfitting issue given limited data

E2Efold (our approach)

- Enforce constraints by using an unrolled algorithm in the architecture
- Restrict the output space

Use Unrolled Algorithms to Enforce Constraints

$$\begin{tikzcd} \boldsymbol{x} = (x_1, x_2, \ldots, x_L) \arrow[r, "\text{Transformer \& Convolution}"] & \boldsymbol{U_\theta}(\boldsymbol{x}) \arrow[r, "\text{Unrolled Algorithm}"] & \boldsymbol{A}^* \end{tikzcd}$$

defines

Constrained optimization

$$\max_{\boldsymbol{A}\in[\mathbf{0},\mathbf{1}]^{L\times L}} \frac{1}{2}\langle \boldsymbol{U_\theta}(\boldsymbol{x}), A\rangle - \boldsymbol{\rho}\|\boldsymbol{A}\|_{\mathbf{1}}$$

$$\text{s.t.} \quad \begin{aligned} M(x) \circ A &= A \\ A^{\top} &= A \\ A\mathbf{1} &\leq \mathbf{1} \\ A &\geq 0 \end{aligned}$$

formulated

- Only {A - U, C - G, G - U} are valid pairings.
- No sharp loops are allowed.
- No overlap of pairs is allowed, i.e., it is a matching.

Use Unrolled Algorithms to Enforce Constraints

Equivalent unconstrained form

$$\mathcal{T}\left(\hat{A}\right) \coloneqq \frac{1}{2}\left(\hat{A} \circ \hat{A} + \left(\hat{A} \circ \hat{A}\right)^{\top}\right) \circ M(\boldsymbol{x})$$

$$\min_{\lambda} \max_{\hat{A} \in [0,1]^{L \times L}} \underbrace{\frac{1}{2}\left\langle \boldsymbol{U}_{\boldsymbol{\theta}}(\boldsymbol{x}), \mathcal{T}(\hat{A})\right\rangle - \left\langle \boldsymbol{\lambda}, \mathrm{relu}\left(\mathcal{T}\left(\hat{A}\right)\mathbf{1} - \mathbf{1}\right)\right\rangle}_{\coloneqq f(\boldsymbol{x}, \hat{A}, \lambda)} - \rho\left\|\hat{A}\right\|_{1}$$

Use Unrolled Algorithms to Enforce Constraints

Equivalent unconstrained form

$$\mathcal{T}(\hat{A}) \coloneqq \frac{1}{2}\left(\hat{A} \circ \hat{A} + \left(\hat{A} \circ \hat{A}\right)^{\top}\right) \circ M(\boldsymbol{x})$$

$$\min_{\lambda} \max_{\hat{A} \in \{0,1\}^{L \times L}} \underbrace{\frac{1}{2}\left\langle \boldsymbol{U}_{\boldsymbol{\theta}}(\boldsymbol{x}), \mathcal{T}(\hat{A}) \right\rangle - \left\langle \boldsymbol{\lambda}, \mathrm{relu}\left(\mathcal{T}\left(\hat{A}\right)\mathbf{1} - \mathbf{1}\right) \right\rangle}_{\coloneqq f(\boldsymbol{x}, \hat{A}, \lambda)} - \rho \left\|\hat{A}\right\|_1$$

Algorithm for solving it (primal-dual)

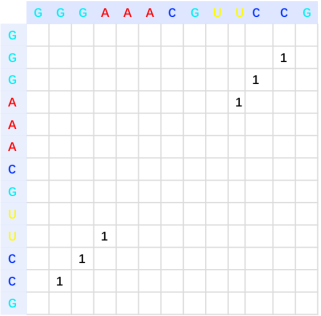
(primal update) $\hat{A} \leftarrow \eta_{\rho\alpha\gamma_{\alpha}^{t}}\left(\hat{A} + \alpha * \gamma_{\alpha}^{t} * \nabla_{\hat{A}} f(x, \hat{A}, \lambda)\right)$

(dual update) $\lambda \leftarrow \lambda + \beta * \gamma_{\beta}^{t} * \mathrm{relu}\left(\mathcal{T}\left(\hat{A}\right)\mathbf{1} - \mathbf{1}\right)$

$\hat{A},\ \lambda$

Until convergence →

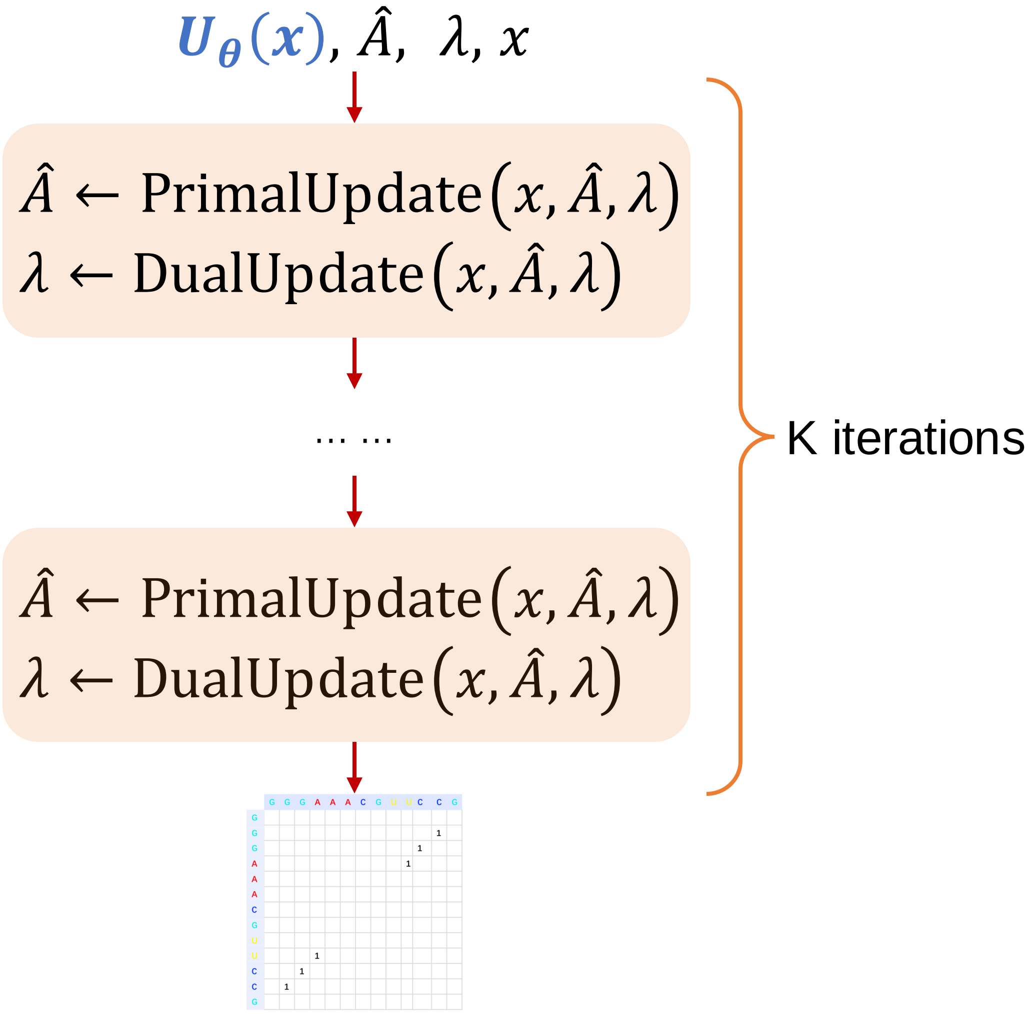
Constraints are satisfied!



	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

Use Unrolled Algorithms to Enforce Constraints

Unrolled Algorithm



$U_\theta(x), \hat{A}, \ \lambda, x$

$\hat{A} \leftarrow \text{PrimalUpdate}(x, \hat{A}, \lambda)$
$\lambda \leftarrow \text{DualUpdate}(x, \hat{A}, \lambda)$

… …

$\hat{A} \leftarrow \text{PrimalUpdate}(x, \hat{A}, \lambda)$
$\lambda \leftarrow \text{DualUpdate}(x, \hat{A}, \lambda)$

K iterations

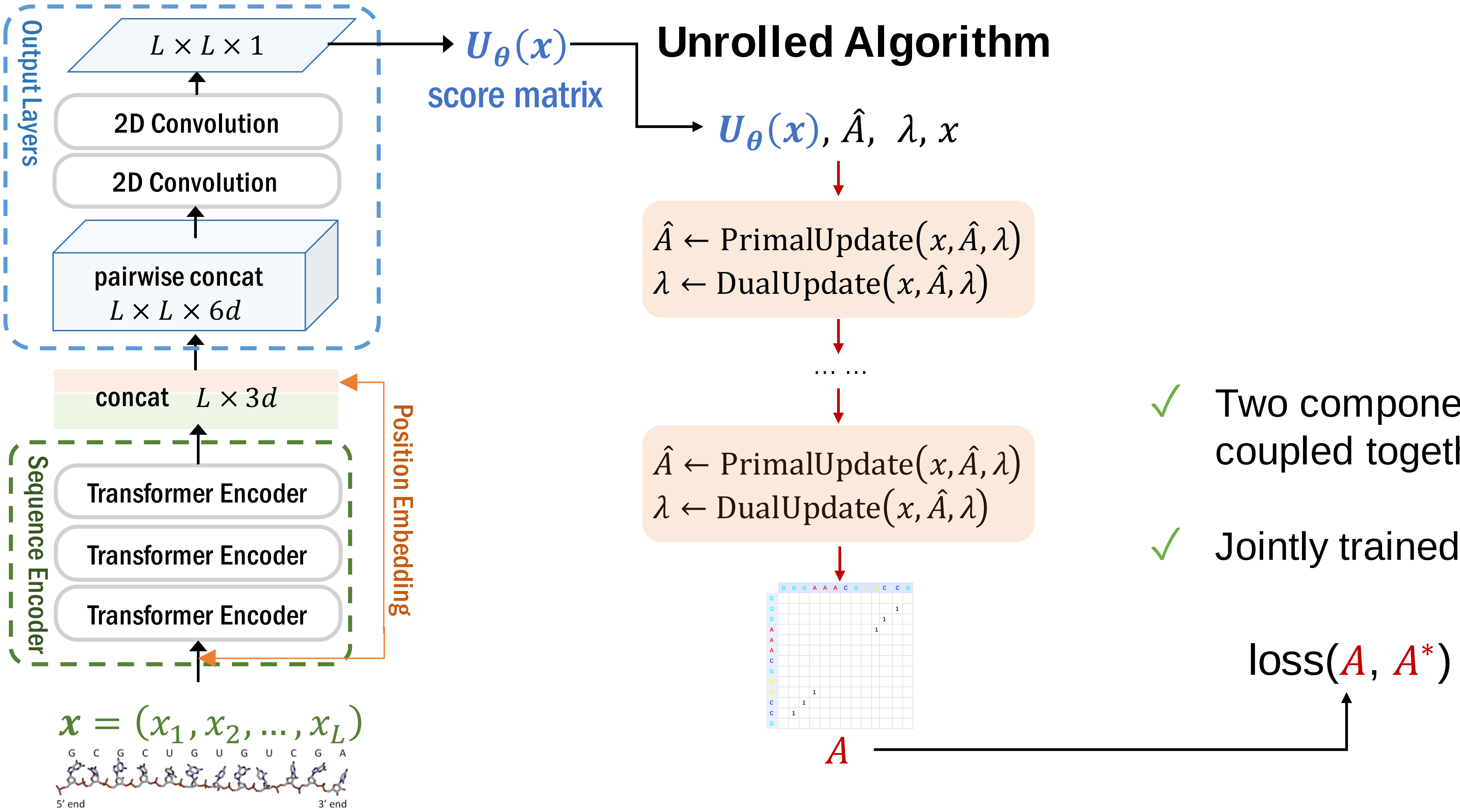
	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

Unrolled Algorithm as Neural Network

each iteration → a recurrent cell
number of iterations → number of layers
hyperparameters → learnable parameters

✓ More structured
✓ Constraints can be gradually enforced

The Overall Model of E2Efold



Output Layers

$L \times L \times 1$

2D Convolution

2D Convolution

pairwise concat $L \times L \times 6d$

concat $L \times 3d$

Position Embedding

Sequence Encoder

Transformer Encoder

Transformer Encoder

Transformer Encoder

$\boldsymbol{x} = (x_1, x_2, \ldots, x_L)$

$\boldsymbol{U_\theta(x)}$ score matrix

Unrolled Algorithm

$\boldsymbol{U_\theta(x)}, \hat{A}, \lambda, x$

$\hat{A} \leftarrow \mathrm{PrimalUpdate}(x, \hat{A}, \lambda)$
$\lambda \leftarrow \mathrm{DualUpdate}(x, \hat{A}, \lambda)$

… …

$\hat{A} \leftarrow \mathrm{PrimalUpdate}(x, \hat{A}, \lambda)$
$\lambda \leftarrow \mathrm{DualUpdate}(x, \hat{A}, \lambda)$

	G	G	G	A	A	A	C	G	U	U	C	C	G
G													
G												1	
G											1		
A										1			
A													
A													
C													
G													
U													
U				1									
C			1										
C		1											
G													

A

loss(A, A^*)

✓ Two component are coupled together

✓ Jointly trained

Differentiable F1 Loss

- F1, precision, recall are commonly used evaluation metric
- But not differentiable.

- We define the following differentiable functions on $[0,1]^{L\times L}$

$$\begin{aligned} &\text{True Positive } = \langle A, A^* \rangle, && \text{False Positive } = \langle A, 1 - A^* \rangle \\ &\text{False Negative } = \langle 1 - A, A^* \rangle, && \text{True Negative } = \langle 1 - A, 1 - A^* \rangle \end{aligned}$$

- $\mathrm{F1} \coloneqq 2\langle A, A^* \rangle / (2\langle A, A^* \rangle + \langle A, 1 - A^* \rangle + \langle 1 - A, A^* \rangle)$
- Directly optimize F1 score!
- Automatically handle the label-imbalanced (more negative samples) issue!

Overall Performance

RNAStralign data: 30451 RNAs from 8 families

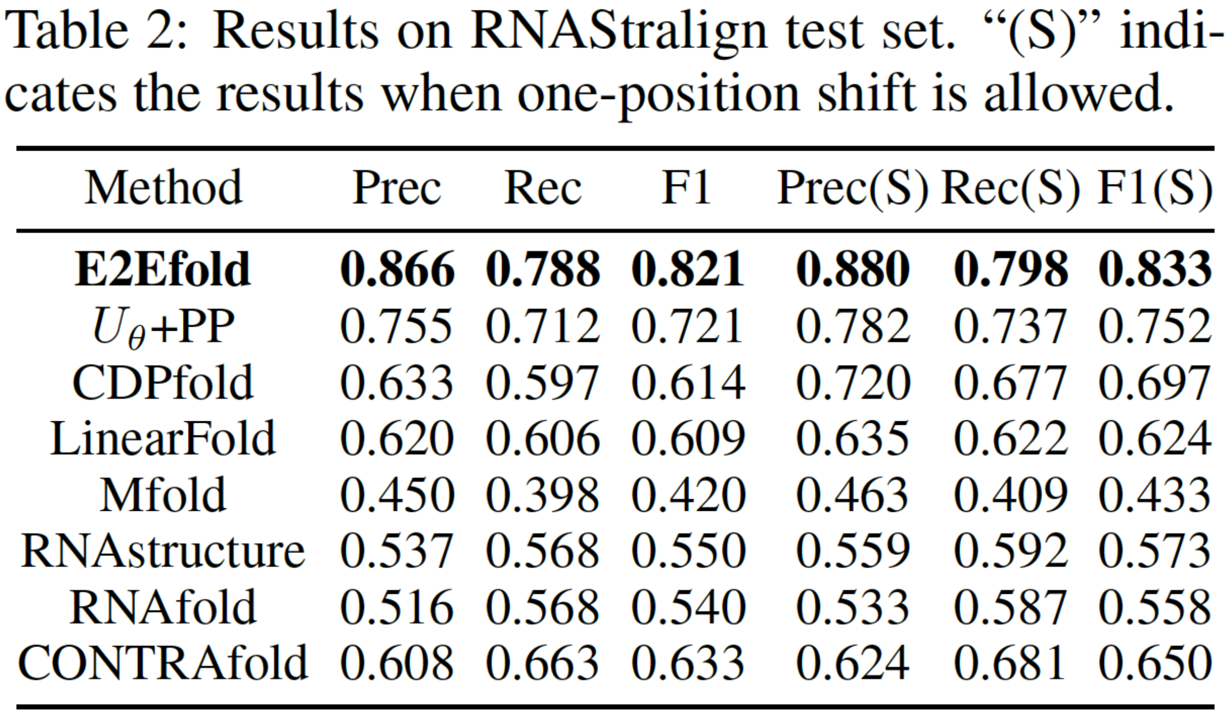


Table 2: Results on RNAStralign test set. “(S)” indicates the results when one-position shift is allowed.

Method	Prec	Rec	F1	Prec(S)	Rec(S)	F1(S)
E2Efold	**0.866**	**0.788**	**0.821**	**0.880**	**0.798**	**0.833**
U_θ+PP	0.755	0.712	0.721	0.782	0.737	0.752
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

Around **20%** improvement

Pseudoknot Prediction

On RNAStralign dataset

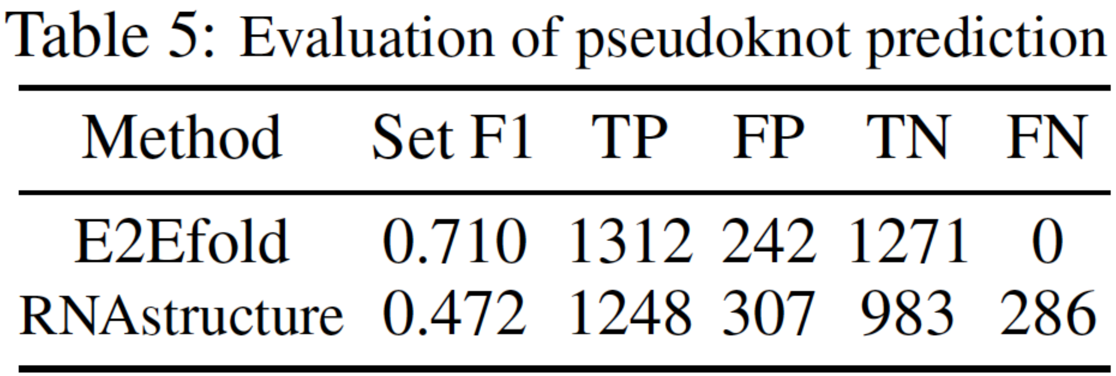
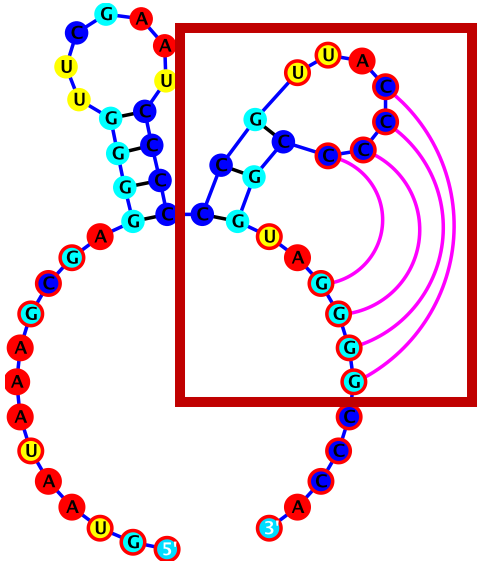


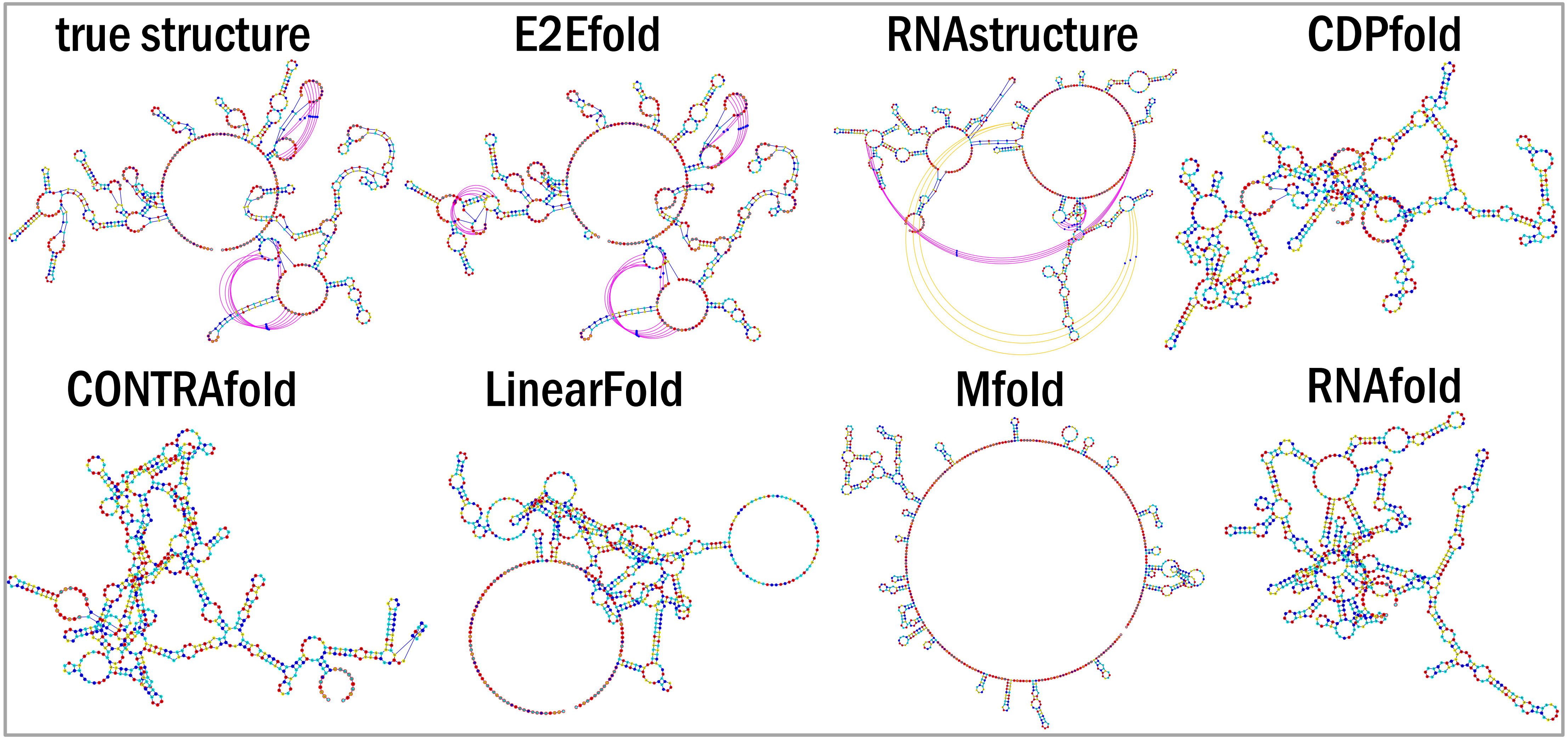
Table 5: Evaluation of pseudoknot prediction

Method	Set F1	TP	FP	TN	FN
E2Efold	0.710	1312	242	1271	0
RNAstructure	0.472	1248	307	983	286



25% improvement on pseudoknot prediction

Visualization of Predicted Structures



true structure

E2Efold

RNAstructure

CDPfold

CONTRAfold

LinearFold

Mfold

RNAfold

Inference Efficiency

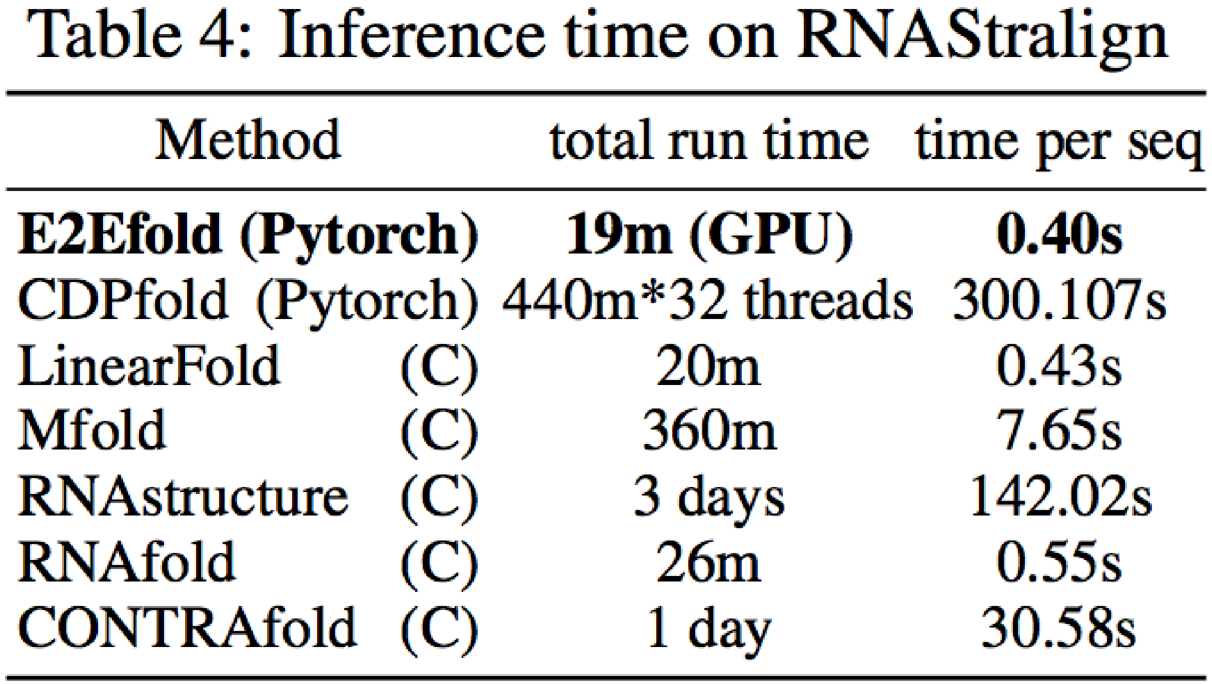
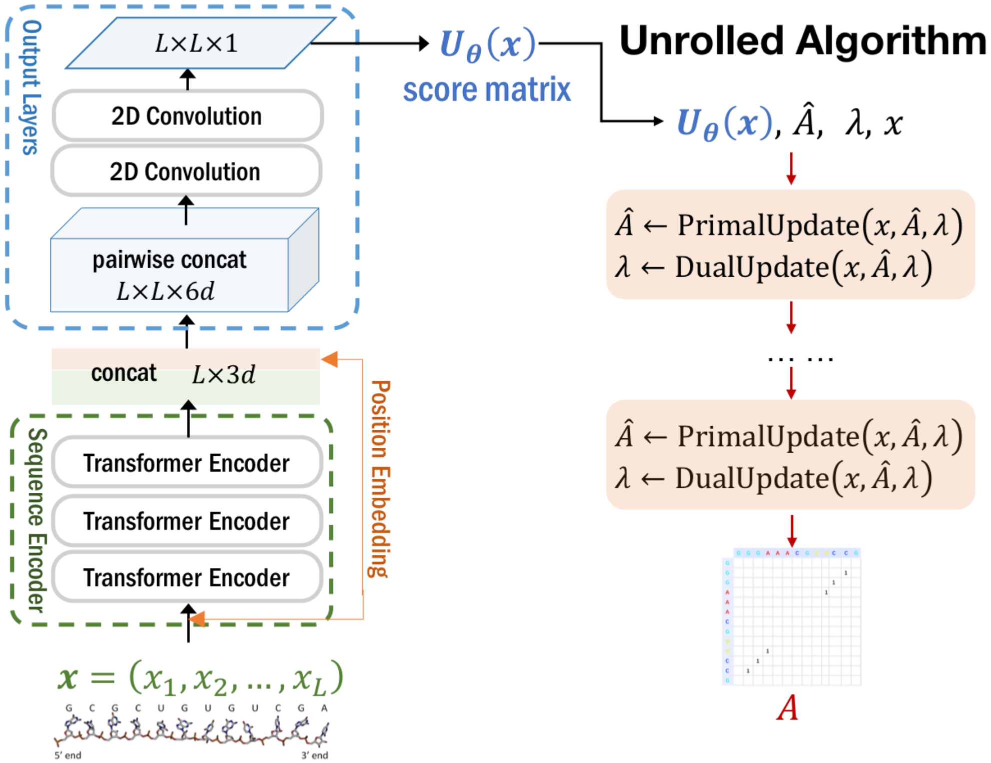


Table 4: Inference time on RNAStralign

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

Conclusion



Output Layers
$L \times L \times 1$
2D Convolution
2D Convolution
pairwise concat
$L \times L \times 6d$
concat $L \times 3d$
Position Embedding
Sequence Encoder
Transformer Encoder
Transformer Encoder
Transformer Encoder
$\boldsymbol{x} = (x_1, x_2, \dots, x_L)$
G C G C U G U G U C G A
5' end
3' end
$\boldsymbol{U_\theta(x)}$
score matrix
Unrolled Algorithm
$\boldsymbol{U_\theta(x)}, \hat{A}, \lambda, x$
$\hat{A} \leftarrow \mathrm{PrimalUpdate}(x, \hat{A}, \lambda)$
$\lambda \leftarrow \mathrm{DualUpdate}(x, \hat{A}, \lambda)$
… …
$\hat{A} \leftarrow \mathrm{PrimalUpdate}(x, \hat{A}, \lambda)$
$\lambda \leftarrow \mathrm{DualUpdate}(x, \hat{A}, \lambda)$
A

- Unrolled algorithm to incorporate constraints in deep architecture design

- SOTA performance in RNA structure prediction, especially for pseudoknots

- Same strategy can be applied to other structured prediction problems
  - NLP (e.g., parsing)
  - CV (e.g., matching)

Paper https://openreview.net/forum?id=S1eALyrYDH

Github code https://github.com/ml4bio/e2efold