Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Structural Bioinformatics

ConsAlign: simultaneous RNA structural aligner based on rich transfer learning and thermodynamic ensemble model of alignment scoring

Masaki Tagashira [1,*]

[1] Department of Computational Biology and Medical Sciences, University of Tokyo, Chiba 277-8561, Japan

* To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: To capture structural homology in RNAs, predicting RNA structural alignments has been a fundamental framework around RNA science. Learning simultaneous RNA structural alignments in their rich scoring parameterization is an undeveloped subject because evaluating them is computationally expensive in nature.

Results: We developed ConsTrain—a gradient-based machine learning method for rich structural alignment scoring. We also implemented ConsAlign—a simultaneous RNA structural aligner composed of ConsTrain's learned scoring parameters. To aim for better structural alignment quality, ConsAlign employs (1) transfer learning from well-defined scoring models and (2) the ensemble model between the ConsTrain model and a mature thermodynamic scoring model. Keeping comparable running time, ConsAlign demonstrated the best alignment prediction quality among current RNA structural aligners.

Availability and implementation: Our code and our data are freely available at `https://github.com/heartsh/consalign`.

Contact: heartsh@heartsh.io

Supplementary information: Supplementary data are available at *Bioinformatics*

1 Introduction

Machine learning is applied to each significant bioinformatics subject, including structural bioinformatics. Also, machine learning has attracted computational scientists' attention around folding single RNA sequences—or *mono-folding*. In mono-folding, modern parameter training methods optimize secondary structure scoring parameters based on both training RNA sequences and training RNA secondary structures (Do *et al*., 2006b; Zakov *et al*., 2011; Rivas *et al*., 2012; Singh *et al*., 2019; Sato *et al*., 2021). Folding RNA homolog sequences and aligning them at the same time—or *simultaneous RNA structural alignment* (Sankoff, 1985)—are an extended solution of mono-folding. Simultaneous RNA structural alignment improves the folding quality of a focused RNA homolog sequence, incorporating RNA homolog sequences concerned. The machine learning methods of simultaneous RNA structural alignment have been developed less initiatively than those of mono-folding due to the complexity of the former (Dowell and Eddy, 2006; Do *et al*., 2008).

We propose *ConsTrain*—a *sparse scoring learning extension* of our sparse inside-outside algorithm (Tagashira and Asai, 2022) in this study (Fig. 1). In Tagashira and Asai (2022), our sparse inside-outside algorithm scores possible simultaneous structural alignments by the scoring parameters of (1) Turner's (nearest neighbor physics) model for RNA secondary structures (Turner and Mathews, 2010) and (2) the CONTRAlign model for RNA sequence alignments (Do *et al*., 2006b). ConsTrain resolves that combining these two independent models is not the optimum solution for scoring possible RNA structural alignments. ConsTrain is based on the *pairwise conditional log-linear model* (= pair-CLLM)—a flexible discriminative training framework (Do *et al*., 2006a). Scoring parameter training for simultaneous RNA structural alignment has been proposed using not the pair-CLLM but the pairwise stochastic context-free grammar (= the pair-SCFG)—a generative parameter training framework (Dowell and Eddy, 2006). Regarding mono-folding accuracy, CLLMs are generally superior to SCFGs due to SCFGs' model constraints (Do *et al*., 2006a). To test the prediction performance of ConsTrain's trained alignment parameters, we build *ConsAlign*—a simultaneous RNA structural aligner parameterized with ConsTrain's

© The Author 2015.

trained alignment parameters. ConsAlign conducts sparse dynamic programming according to the *progressive alignment scheme* (Feng and Doolittle, 1987). ConsAlign depends on its hyper-parameters to control simultaneous structural alignment quality but corrects them automatically without exploiting any validation data.

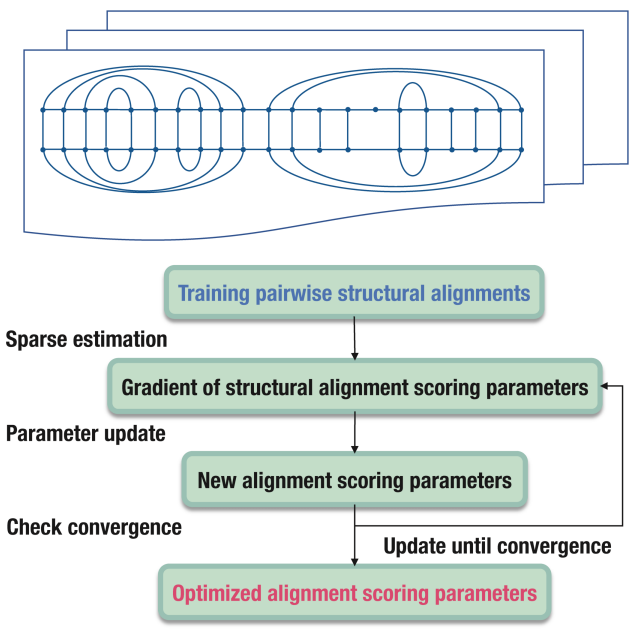

Fig. 1. Our proposed workflow of ConsTrain. From a training dataset composed of RNA pairwise structural alignments, ConsTrain iteratively learns scoring parameters for simultaneous RNA structural alignment until they converge.

2 Methods

2.1 Computing alignment scoring parameters' gradient

We can optimize the scoring parameters of any CLLM (including our structural alignment pair-CLLM), given their cost function (Do *et al*., 2006a; Andronescu *et al*., 2007). From both the empirical occurrence counts and the expected occurrence counts of pair-CLLM scoring parameters in observed pairwise structural alignments, we can derive the *gradient* of a pair-CLLM cost function. By conducting our inside-outside algorithm on sparse pairwise structural alignment (Tagashira and Asai, 2022) iteratively, we compute the approximated expected counts of scoring parameters appearing in training pairwise structural alignments.

2.1.1 Forming the cost of training pairwise structural alignments

We derive a *convex* pair-CLLM cost function fully parameterized with tunable alignment scoring parameters from training pairwise structural alignments. We score any pairwise structural alignment $\mathbb{A}$ by using a simple inner product:

$$s(\mathbb{A};\boldsymbol{\theta}) \stackrel{\text{def}}{=} \boldsymbol{\theta}^{\mathrm{T}} \cdot \boldsymbol{\phi}(\mathbb{A})$$

Here, $\boldsymbol{\theta}$ is any scoring parameter vector in the real F-space. In addition, a function $\boldsymbol{\phi}(\mathbb{A})$ maps any pairwise structural alignment $\mathbb{A}$ to the vector that counts the occurrence of each f-th scoring parameter $\theta_f : (\theta_f) \equiv \boldsymbol{\theta}, f \in \{1,\ldots,F\}$ in $\mathbb{A}$. The pairwise structural alignment scoring $s(\mathbb{A};\boldsymbol{\theta})$ obeys some pair-CLLM. As a result, each alignment scoring parameter θ_f does not have to take the form of a log probability nor correspond only to a transition/emission (Do *et al*., 2006a).

Onto the training structural alignment $\mathbb{A}_d$ of each d-th RNA sequence pair, we obtain some convex pair-CLLM cost (Do *et al*., 2006a; Andronescu *et al*., 2007)

$$\begin{aligned} c(D;\boldsymbol{\theta}) &\stackrel{\text{def}}{=} -\sum_{d:d\in\{1,\ldots,|D|\}} \ln \frac{\mathrm{e}^{s(\mathbb{A}_d;\boldsymbol{\theta})}}{Z_d(\boldsymbol{\theta})} \\ &\equiv \sum_{d:d\in\{1,\ldots,|D|\}} [\ln Z_d(\boldsymbol{\theta}) - s(\mathbb{A}_d;\boldsymbol{\theta})]. \end{aligned}$$

Here, any training alignment dataset D is $D \stackrel{\text{def}}{=} \{\mathbb{A}_d\}$. Furthermore, any alignment *partition function* $Z_d(\boldsymbol{\theta})$ is $Z_d(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d} \mathrm{e}^{s(\mathbb{A}';\boldsymbol{\theta})}$. ($\mathcal{A}_d$ is the structural alignment *Boltzmann ensemble*—i.e., a set of all possible structural alignments—of each d-th sequence pair.) Some convex pair-CLLM cost $c(D;\boldsymbol{\theta})$ holds its gradient

$$\nabla c(D;\boldsymbol{\theta}) \stackrel{\text{def}}{=} (\frac{\partial c(D;\boldsymbol{\theta})}{\partial \theta_f}) \equiv \sum_{d:d\in\{1,\ldots,|D|\}} \{\mathbb{E}_d[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}] - \boldsymbol{\phi}(\mathbb{A}_d)\}$$

because any log function gradient $(\frac{\partial \ln Z_d(\boldsymbol{\theta})}{\partial \theta_f})$ is

$$\begin{aligned} (\frac{\partial \ln Z_d(\boldsymbol{\theta})}{\partial \theta_f}) &\equiv (\frac{1}{Z_d(\boldsymbol{\theta})} \frac{\partial \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d} \mathrm{e}^{s(\mathbb{A}';\boldsymbol{\theta})}}{\partial \theta_f}) \\ &\equiv \frac{1}{Z_d(\boldsymbol{\theta})} \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d} \boldsymbol{\phi}(\mathbb{A}')\mathrm{e}^{s(\mathbb{A}';\boldsymbol{\theta})} \qquad (1) \\ &\equiv \mathbb{E}_d[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}]. \end{aligned}$$

Here, any expected count vector $\mathbb{E}_d[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}]$ is

$$\mathbb{E}_d[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}] \stackrel{\text{def}}{=} \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d} p_d(\mathbb{A}';\boldsymbol{\theta})\boldsymbol{\phi}(\mathbb{A}') \equiv \text{Equation (1)}.$$

[Any alignment *Boltzmann (probability) distribution* $p_d(\mathbb{A}';\boldsymbol{\theta})$ is $p_d(\mathbb{A}';\boldsymbol{\theta}) \stackrel{\text{def}}{=} \frac{\mathrm{e}^{s(\mathbb{A}';\boldsymbol{\theta})}}{Z_d(\boldsymbol{\theta})}$.]

By using our sparse inside-outside algorithm, we can estimate the approximate amount of any expected count vector $\mathbb{E}_d[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}]$ (Do *et al*., 2006a; Andronescu *et al*., 2007):

$$\mathbb{E}_d[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}] \equiv \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d} p_d(\mathbb{A}';\boldsymbol{\theta})\boldsymbol{\phi}(\mathbb{A}') \approx \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d^*} p_d^*(\mathbb{A}';\boldsymbol{\theta})\boldsymbol{\phi}(\mathbb{A}').$$

Here, $\mathcal{A}_d^*$ is a sparse version of any alignment Boltzmann ensemble $\mathcal{A}_d$. In addition, any sparse Boltzmann distribution $p_d^*(\mathbb{A}';\boldsymbol{\theta})$ is as follows:

$$\begin{aligned} \mathbb{A}' \sim p_d(\mathbb{A}';\boldsymbol{\theta}) \approx p_d^*(\mathbb{A}';\boldsymbol{\theta}) &\stackrel{\text{def}}{=} \frac{\mathrm{e}^{s(\mathbb{A}';\boldsymbol{\theta})}}{Z_d^*(\boldsymbol{\theta})} \\ Z_d(\boldsymbol{\theta}) \approx Z_d^*(\boldsymbol{\theta}) &\stackrel{\text{def}}{=} \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d^*} \mathrm{e}^{s(\mathbb{A}';\boldsymbol{\theta})}. \end{aligned}$$

This approximated vector estimation requires the same running time $O(L^2)$ and memory usage $O(L^2)$ as our sparse inside-outside algorithm (Tagashira and Asai, 2022). Here, L is the longer length of each d-th sequence pair.

2.2 Optimizing structural alignment scoring parameters

Our parameter optimization goal is to find structural alignment scoring parameters that minimize their convex pair-CLLM cost. We can utilize general gradient-based optimization methods to complete our scoring parameter optimization. We aim to compute some optimized parameter

vector

$$\boldsymbol{\theta}^* \stackrel{\text{def}}{=} \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^F} c^*(D;\boldsymbol{\theta})$$

$$c^*(D;\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{d:d\in\{1,\ldots,|D|\}} [\ln Z_d^*(\boldsymbol{\theta}) - s(\mathbb{A}_d;\boldsymbol{\theta})]$$

(where $\mathbb{R}^F$ is the real F-space) using the gradient of any approximated pair-CLLM cost $c^*(D;\boldsymbol{\theta})$

$$\nabla c^*(D;\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{d:d\in\{1,\ldots,|D|\}} \{\mathbb{E}_d^*[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}] - \boldsymbol{\phi}(\mathbb{A}_d)\}$$

$$\mathbb{E}_d^*[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}] \stackrel{\text{def}}{=} \sum_{\mathbb{A}':\mathbb{A}'\in\mathcal{A}_d^*} p_d^*(\mathbb{A}';\boldsymbol{\theta})\boldsymbol{\phi}(\mathbb{A}').$$

By Algorithm 1, we can conduct the above convex parameter optimization iteratively. Algorithm 1 is an instance of the Broyden-Fletcher-Goldfarb-Shanno (= BFGS) algorithm—a quasi-Newton optimization algorithm widely used (Fletcher, 1987).

Algorithm 1 BFGS algorithm that minimizes any approximated convex pair-CLLM cost $c^*(D;\boldsymbol{\theta})$.

function bfgsAlgo(a training structural alignment dataset D)
  Initialize structural alignment scoring parameters $\boldsymbol{\theta}^*$
  for $d \in \{1,\ldots,|D|\}$ **do**
    Get parameter counts $\boldsymbol{\phi}(\mathbb{A}_d)$ by parsing each training structural alignment $\mathbb{A}_d$
  while $\boldsymbol{\theta}^*$ does not converge **do**
    for $d \in \{1,\ldots,|D|\}$ **do**
      Compute expected counts $\mathbb{E}_d^*[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}^*]$ using our inside-outside algorithm
    $\nabla c^*(D;\boldsymbol{\theta}^*) \leftarrow \sum_{d:d\in\{1,\ldots,|D|\}}\{\mathbb{E}_d^*[\boldsymbol{\phi}(\cdot);\boldsymbol{\theta}^*] - \boldsymbol{\phi}(\mathbb{A}_d)\}$
    /* Apply the parameter update scheme of the BFGS algorithm */
    $\boldsymbol{\theta}^* \leftarrow \text{argmin}^{\text{BFGS}}[c^*(D;\boldsymbol{\theta}^*)|\nabla c^*(D;\boldsymbol{\theta}^*)]$
  return $\boldsymbol{\theta}^*$

Any approximated pair-CLLM cost $c^*(D;\boldsymbol{\theta})$ does not retain any regularization term. In other words, some optimized parameter vector $\boldsymbol{\theta}^*$ is prone to be overfitted for any training alignment dataset D. Thus, we minimize some L2-regularized non-convex cost instead of some unregularized convex cost $c^*(D;\boldsymbol{\theta})$:

$$\boldsymbol{\theta}^* \stackrel{\text{def}}{=} \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^F} c^{\text{R}}(D;\boldsymbol{\theta},\alpha,\beta)$$

$$c^{\text{R}}(D;\boldsymbol{\theta},\alpha,\beta) \stackrel{\text{def}}{=} c^*(D;\boldsymbol{\theta}) + \frac{1}{2}\boldsymbol{\lambda}^{\text{T}}\cdot(\theta_f^2)$$

$$\boldsymbol{\lambda} \stackrel{\text{def}}{=} \left(\frac{\frac{|G(f)|}{2}+\alpha}{\beta+\frac{1}{2}\sum_{f':f'\in\{1,\ldots,F\}|G(f')=G(f)}\theta_{f'}^2}\right).$$

Here, α,β are the two gamma distribution parameters. Also, a function $G(f)$ returns the group (set) where each alignment scoring parameter θ_f belongs. Any adaptive L2-regularization vector $\boldsymbol{\lambda}$ introduces an alignment parameter prior (probability) distribution:

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta};\alpha,\beta) \propto \Pi_{f=1}^F \Gamma\left(\frac{|G(f)|}{2}+\alpha\right)\Sigma_f(\boldsymbol{\theta};\beta)^{-\frac{|G(f)|}{2}-\alpha}$$

$$\Sigma_f(\boldsymbol{\theta};\beta) \stackrel{\text{def}}{=} \beta + \frac{1}{2}\sum_{f':f'\in\{1,\ldots,F\}|G(f')=G(f)}\theta_{f'}^2.$$

Here, the gamma function $\Gamma(r)$ given any real number r is $\Gamma(r) \stackrel{\text{def}}{=} \int_{t=0}^{\infty} t^{r-1}\text{e}^{-t}dt$. By Algorithm S1 based on Algorithm 1, we implement the above non-convex parameter optimization iteratively. Algorithm S1 is an example of *majorization-minimization algorithms*—iterative optimization algorithms that minimize some convex upper bound of any non-convex cost function (Langed *et al*., 2000). Algorithm S1 groups any alignment scoring parameter θ_f into clusters and penalizes each scoring parameter cluster by a shared regularization constant (Foo *et al*, 2009). The advantage of introducing some parameter prior distribution $p(\boldsymbol{\theta};\alpha,\beta)$ is that we do not have to predetermine the value of any L2-regularization constant vector $\boldsymbol{\lambda}$ (e.g., by a grid search upon a validation structural alignment dataset) since we (in addition to any alignment scoring parameter $\boldsymbol{\theta}$) optimize $\boldsymbol{\lambda}$ in the same procedure (Foo *et al*, 2009). We implemented Algorithm S1 as ConsTrain version 0.1.0 (available from `https://github.com/heartsh/consprob-trained`).

2.3 Configuring structural alignment scoring parameters

Borrowing knowledge from conventional scoring parameter optimizations, we specify the concrete settings of our alignment scoring parameters. We specify all the groups (or types) of our alignment scoring parameters, as described in Table S1. The *CONTRAfold model* is a CLLM that (from the secondary structure of each single RNA sequence) learns secondary structure scoring parameters based on RNA loop structures (Do *et al*., 2006a). The *CONTRAlign model* is a pair-conditional random field (Lafferty *et al*., 2001; Sato and Sakakibara, 2005) that trains sequence alignment scoring parameters from each sequence-aligned RNA pair (Do *et al*., 2006b). Our pair-CLLM inherits scoring parameter formats from both the CONTRAfold and the CONTRAlign models.

2.4 Our proposed structural aligner to maximize accuracy

2.4.1 Quadratic pairwise alignment guaranteeing its optimality

In the γ-centroid estimator principle (Ding *et al*., 2005; Hamada *et al*., 2009a,b,c, 2011), we define our pairwise structural alignment as the following plain recursion like LocARNA (Will *et al*., 2007) and RAF (Do *et al*., 2008):

$$a_{ijkl} \equiv b_{ik}(j-1,l-1) + \alpha_{ijkl}$$

$$b_{ik}(j,l) \equiv \max\begin{cases} b_{ik}(j-1,l) \\ b_{ik}(j,l-1) \\ b_{ik}(j-1,l-1) + \gamma^{\text{M}}p_{jl}^{\text{M}}(\boldsymbol{\theta}^*) - 1 \\ \max_{uv:i<u<j,k<v<l}[b_{ik}(u-1,v-1) + a_{ujvl}] \end{cases}$$

$$\alpha_{ijkl} \stackrel{\text{def}}{=} \gamma^{\text{M}}[p_{ik}^{\text{M}}(\boldsymbol{\theta}^*) + p_{jl}^{\text{M}}(\boldsymbol{\theta}^*)] + \gamma^{\text{P}}[p_{ij}^{\text{P}}(\boldsymbol{\theta}^*) + p_{kl}^{\text{P}}(\boldsymbol{\theta}^*)] - 4$$

starting from the initial condition

$$b_{ik}(i,k) \leftarrow 0.$$

Here, $p_{ik}^{\text{M}}(\boldsymbol{\theta}^*)$ is the sparse posterior probability that any nucleotide (position) pair (i,k) matches in pairwise structural alignment. Furthermore, $p_{ij}^{\text{P}}(\boldsymbol{\theta}^*)$ is the average sparse probabilistic consistency that every two nucleotides i,j base-pair in pairwise structural alignment. ConsProb (Tagashira and Asai, 2022) parameterized with our trained scoring parameters $\boldsymbol{\theta}^*$ can compute any sparse probabilities $p_{ik}^{\text{M}}(\boldsymbol{\theta}^*), p_{ij}^{\text{P}}(\boldsymbol{\theta}^*)$. Also, $\gamma^{\text{M}},\gamma^{\text{P}}$ are some two real numbers greater than 1. Some hyper-parameters γ^{M} and γ^{P} control the predictive significance of nucleotide matching and nucleotide base-pairing, respectively.

In a naive sense, we can conduct our pairwise predictive recursion with both $O(N^3M^3)$ running time and $O(N^2M^2)$ memory usage. (N,M are the lengths of every two RNA sequences.) However, we can perform our pairwise predictive recursion only with both $O(L^2)$ running time and $O(L)$ memory usage if some hyper-parameters $\gamma^{\text{M}},\gamma^{\text{P}}$ are sufficiently small. [L is $L \stackrel{\text{def}}{=} \max(N,M)$.] We can realize this computational

complexity reduction because our pairwise predictive recursion accepts only some nucleotide matches (i, k) satisfying the conditions

$$p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*) \geq \frac{1}{\gamma^{\mathrm{M}}} \tag{2}$$

and some nucleotide base-pairings $(i, j), (k, l)$ satisfying the conditions

$$p^{\mathrm{P}}_{ij}(\boldsymbol{\theta}^*) \geq \frac{1}{\gamma^{\mathrm{P}}}, p^{\mathrm{P}}_{kl}(\boldsymbol{\theta}^*) \geq \frac{1}{\gamma^{\mathrm{P}}}. \tag{3}$$

LocARNA and RAF impose Equations (2) and (3) upon possible pairwise structural alignments to enable both $O(L^2)$ running time and $O(L)$ memory usage. Our pairwise predictive recursion assures that its obtained structural alignment is optimum (i.e., retaining the best score of all possible structural alignments) even if we impose Equations (2) and (3) upon possible pairwise structural alignments.

2.4.2 Extending pairwise alignment into multiple alignment

Our multiple structural alignment is based on the progressive alignment scheme (Feng and Doolittle, 1987) like conventional RNA structural aligners (Will *et al*., 2007; Do *et al*., 2008; Sato *et al*., 2012; Will *et al*., 2015; Li *et al*., 2021). When we build a *UPGMA guide tree* (Sneath and Sokal, 1962) from RNA homolog sequences to be aligned, we measure the similarity between every two RNA homologs

$$\sigma(\boldsymbol{\theta}^*) \stackrel{\text{def}}{=} \sum_{ik} p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*) I[p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*) \geq \frac{1}{\gamma^{\mathrm{M}}}].$$

Here, $I[c]$ returns 1 if any condition c is true; otherwise, $I[c]$ returns 0. Given some hyper-parameter γ^{M}, our proposed metric $\sigma(\boldsymbol{\theta}^*)$ estimates the stochastic similarity between every two RNA homologs without aligning them. Our proposed metric $\sigma(\boldsymbol{\theta}^*)$ considers only some nucleotide matches (i, k) satisfying the condition $p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*) \geq \frac{1}{\gamma^{\mathrm{M}}}$ to be eligible.

We extend our pairwise predictive recursion into a multiple predictive recursion, replacing any nucleotide matching probability $p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*)$ with any column matching probability

$$p^{\mathrm{M}}_{IK}(\boldsymbol{\theta}^*; X, Y) \stackrel{\text{def}}{=} \frac{1}{|X||Y|} \sum_{xy:x\in X, y\in Y} p^{\mathrm{M}}_{I_x K_y}(\boldsymbol{\theta}^*; x, y)$$

and replacing every two base-pairing probabilities $p^{\mathrm{P}}_{ij}(\boldsymbol{\theta}^*), p^{\mathrm{P}}_{kl}(\boldsymbol{\theta}^*)$ with every two probabilities of alignment column base-pairing

$$p^{\mathrm{P}}_{IJ}(\boldsymbol{\theta}^*; X) \stackrel{\text{def}}{=} \frac{1}{|X|} \sum_{x:x\in X} p^{\mathrm{P}}_{I_x J_x}(\boldsymbol{\theta}^*; x)$$

$$p^{\mathrm{P}}_{KL}(\boldsymbol{\theta}^*; Y) \stackrel{\text{def}}{=} \frac{1}{|Y|} \sum_{y:y\in Y} p^{\mathrm{P}}_{K_y L_y}(\boldsymbol{\theta}^*; y).$$

Here, X, Y are every two aligned sets of RNA homolog sequences to be aligned. Moreover, I, J, K, L are every four alignment columns, and I_x is any RNA homolog x's nucleotide stored in I. $p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*; x, y)$ is the explicit form of any nucleotide matching probability $p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*)$ regarding every two RNA homologs x, y; and $p^{\mathrm{P}}_{ij}(\boldsymbol{\theta}^*; x)$ is the explicit form of any average base-pairing probability $p^{\mathrm{P}}_{ij}(\boldsymbol{\theta}^*)$ regarding x.

2.4.3 Auto-correction of structural alignment hyper-parameters

The disadvantage of γ-centroid estimators is that they rely on hyper-parameters (e.g., hyper-parameters $\gamma^{\mathrm{M}}, \gamma^{\mathrm{P}}$ in our structural alignment). In practice, we have to determine hyper-parameters before performing γ-centroid estimators. This study proposes a reasonable way to determine hyper-parameters in performing γ-centroid estimators. The *sum-of-pairs score* (= SPS) of any predicted structural alignment $\mathbb{A}$ against any possible multiple alignment $\mathbb{A}'$ is

$$\sigma(\mathbb{A}, \mathbb{A}') \stackrel{\text{def}}{=} \frac{1}{C} \sum_{xy:x<y} \{ \sum_{ik:(i,k)\in M_{xy}(\mathbb{A})} I[(i,k) \in M_{xy}(\mathbb{A}')] + \sum_{i:i\in I_{xy}(\mathbb{A})} I[i \in I_{xy}(\mathbb{A}')] + \sum_{k:k\in D_{xy}(\mathbb{A})} I[k \in D_{xy}(\mathbb{A}')] \}.$$

Here, any SPS denominator C is

$$C \stackrel{\text{def}}{=} \sum_{xy:x<y} [\sum_{ik:(i,k)\in M_{xy}(\mathbb{A})} 1 + \sum_{i:i\in I_{xy}(\mathbb{A})} 1 + \sum_{k:k\in D_{xy}(\mathbb{A})} 1].$$

In addition, $M_{xy}(\mathbb{A})$ is a set of all aligned nucleotide pairs between every two RNA homologs x, y in any structural alignment $\mathbb{A}$; $I_{xy}(\mathbb{A})$ is a set of all inserted nucleotides between x, y in $\mathbb{A}$; and $D_{xy}(\mathbb{A})$ is a set of all deleted nucleotides between x, y in $\mathbb{A}$.

As a result, we can measure the approximated expected SPS of any predicted structural alignment $\mathbb{A}$ against all possible multiple alignments

$$\mathbb{E}[\sigma(\mathbb{A}, \cdot); \boldsymbol{\theta}^*] \stackrel{\text{def}}{=} \frac{1}{C} \sum_{xy:x<y} [\sum_{ik:(i,k)\in M_{xy}(\mathbb{A})} p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*; x, y) + \sum_{i:i\in I_{xy}(\mathbb{A})} p^{\mathrm{I}}_{i}(\boldsymbol{\theta}^*; x, y) + \sum_{k:k\in D_{xy}(\mathbb{A})} p^{\mathrm{D}}_{k}(\boldsymbol{\theta}^*; x, y)].$$

Here, $p^{\mathrm{I}}_{i}(\boldsymbol{\theta}^*; x, y)$ is $p^{\mathrm{I}}_{i}(\boldsymbol{\theta}^*; x, y) \stackrel{\text{def}}{=} 1 - \sum_{k:1\leq k\leq M} p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*; x, y)$, and $p^{\mathrm{D}}_{k}(\boldsymbol{\theta}^*; x, y)$ is $p^{\mathrm{D}}_{k}(\boldsymbol{\theta}^*; x, y) \stackrel{\text{def}}{=} 1 - \sum_{i:1\leq i\leq N} p^{\mathrm{M}}_{ik}(\boldsymbol{\theta}^*; x, y)$. Finally, we can optimize some hyper-parameters $\gamma^{\mathrm{M}}, \gamma^{\mathrm{P}}$ as well as some predicted structural alignment by enumerating possible values of $\gamma^{\mathrm{M}}, \gamma^{\mathrm{P}}$ as in Algorithm 2. Our alignment hyper-parameter auto-correction is similar to hyper-parameter auto-correction used in mono-folding—called pseudo-expected accuracy optimization (Hamada *et al*., 2010; Sato and Kato, 2022).

Algorithm 2 Algorithm that optimizes some hyper-parameters $\gamma^{\mathrm{M}}, \gamma^{\mathrm{P}}$ together with some predicted structural alignment.

```
function alignerWrapper(RNA homolog sequences to be aligned, structural alignment scoring parameters θ*)
  for i ∈ {i^min, ..., i^max}, j ∈ {j^min, ..., j^max} do
    γ^M ← 2^i + 1, γ^P ← 2^j + 1
    Predict an alignment 𝔸 using the progressive alignment scheme, given γ^M, γ^P
    Compute the expected sum-of-pairs score 𝔼[σ(𝔸, ·); θ*] of 𝔸
    σ^new ← 𝔼[σ(𝔸, ·); θ*]
    if σ^new > σ^curr or σ^curr is unset then
      𝔸* ← 𝔸, γ^M,* ← γ^M, γ^P,* ← γ^P, σ^curr ← σ^new
  return (𝔸*, γ^M,*, γ^P,*)
```

2.4.4 *Ali-folding* post-process of predicted structural alignment

Folding RNA alignment—or ali-folding—is one of the most powerful ways to find the *conserved base-pairing*s in RNA alignment as well as simultaneous RNA structural alignment (Bernhart *et al*., 2008; Seemann *et al*., 2008; Hamada *et al*., 2011; Tagashira and Asai, 2022). As LocARNA adopts RNAalifold (Bernhart *et al*., 2008) as the ali-folding post-processor of LocARNA's predicted structural alignment (Will *et al*., 2007), we adopt ConsAlifold (Tagashira and Asai, 2022) parameterized with our trained alignment parameters as the ali-folding post-processor of our predicted structural alignment.

2.4.5 Ensemble of different alignment scoring models

Combining different stochastic models for a prediction—referred to as a *model ensemble*—has been utilized in mono-folding to achieve more

accurate structure prediction (Aghaeepour and Hoos, 2013; Singh *et al*., 2019). In our structural alignment prediction, we combine our sparse inside-outside algorithm of Tagashira and Asai (2022) and that of this study to create an alignment scoring model ensemble. More specifically, we use any average matching probability

$$p^{\mathrm{M}}_{IK}(\boldsymbol{\theta}^{\mathrm{old}}, \boldsymbol{\theta}^{*}; X, Y) \stackrel{\mathrm{def}}{=} \frac{1}{2}[p^{\mathrm{M}}_{IK}(\boldsymbol{\theta}^{\mathrm{old}}; X, Y) + p^{\mathrm{M}}_{IK}(\boldsymbol{\theta}^{*}; X, Y)]$$

and any average base-pairing probability

$$p^{\mathrm{P}}_{IJ}(\boldsymbol{\theta}^{\mathrm{old}}, \boldsymbol{\theta}^{*}; X) \stackrel{\mathrm{def}}{=} \frac{1}{2}[p^{\mathrm{P}}_{IJ}(\boldsymbol{\theta}^{\mathrm{old}}; X) + p^{\mathrm{P}}_{IJ}(\boldsymbol{\theta}^{*}; X)].$$

Here, $\boldsymbol{\theta}^{\mathrm{old}}$ is some scoring parameter vector of any pairwise structural alignment used in Tagashira and Asai (2022). Our old scoring parameters $\boldsymbol{\theta}^{\mathrm{old}}$ are from both Turner's and the CONTRAlign models, while our trained scoring parameters $\boldsymbol{\theta}^{*}$ are from both the CONTRAfold and the CONTRAlign models. We implemented our RNA structural aligner explained above as ConsAlign (available from `https://github.com/heartsh/consalign`).

2.5 Benchmarking data

We used Rfam version 14.4 for both our alignment parameter training and our software benchmarking test. From Rfam, we collected 3129 structurally aligned families as dataset "origin." The seed reference structural alignment of each collected RNA family retained at most 500 columns and at most 20 sequences. We randomly split dataset "origin" into both equal datasets "train origin" and "test origin." From dataset "train origin," we collected each RNA family whose seed reference consensus structure is not any prediction product [e.g., neither by Pfold (Knudsen and Hein, 2003) nor by RNAalifold (Bernhart *et al*., 2008)] as dataset "train pure." We random-sampled 10 RNA sequences from each RNA family holding at least 10 RNA sequences in dataset "train pure." Then, we retrieved the all-to-all pairwise structural alignments of these 10 RNA sequences, finally obtaining dataset "train pairwise." This random sequence sampling prevents our alignment parameter learning from overfitting due to the overwhelming number of the RNA homologs belonging to the same large RNA family (Do *et al*., 2006a, 2008). By retrieving a seed multiple structural alignment from each RNA family in dataset "test origin," we gained dataset "test multiple."

As a result, (1) dataset "train pairwise" is composed of 2250 pairwise structural alignments, and (2) dataset "test multiple" is composed of 1565 multiple structural alignments. We used dataset "train pairwise" in order to train ConsTrain. We used dataset "test multiple" to assess the prediction performances of each benchmarked prediction method. Any entry in dataset "train pairwise" and any entry in dataset "test multiple" are not derived from the same family to prevent structural homology from leaking between training data and test data (Rivas *et al*., 2012; Sato *et al*., 2021).

2.6 Conventional structural aligners for experiments

Using default command line arguments, we compared (1) RAF version 1.00, (2) LocARNA version 2.0.0, (3) SPARSE version 2.0.0 (Will *et al*., 2015), (4) DAFS version 0.0.3 (Sato *et al*., 2012), and (5) LinearTurboFold version 1 (Li *et al*., 2021) to ConsAlign. (1) RAF (a) is a simultaneous RNA structural aligner taking both CONTRAfold and CONTRAlign's posterior probabilities and (b) trains the weight parameters of these posterior probabilities in max-margin optimization (Do *et al*., 2008). (2, 3) LocARNA and SPARSE are different implementations of Sankoff's algorithm utilizing simultaneous structural alignment sparsity (Will *et al*., 2007, 2015). SPARSE is a variant of LocARNA and exploits only structure-based constraints (Will *et al*., 2015), whereas LocARNA exploits matching-based and structure-based constraints (Will *et al*., 2007). (4) DAFS is a simultaneous RNA structural aligner and realizes reasonable computational complexities by applying dual decomposition to integer programming (Sato *et al*., 2012). (5) LinearTurboFold is an application of LinearPartition (Zhang *et al*., 2020) to TurboFold, an iterative RNA structural aligner (Tan *et al*., 2017). LinearTurboFold realizes its quick predictive iteration using both LinearPartition and beam search-based sequence alignment (Li *et al*., 2021).

2.7 Alignment prediction accuracy metrics

In each RNA family, we find the numbers of true positives, true negatives, false positives, and false negatives TP, TN, FP, and FN to its consensus secondary structure while benchmarking each tool. Per RNA family, we compute an F1 score and a Matthews correlation coefficient (= MCC) as well as an SPS:

$$F1 \stackrel{\mathrm{def}}{=} \frac{2 \times TP}{2 \times TP + FP + FN}$$

$$MCC \stackrel{\mathrm{def}}{=} \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}.$$

2.8 Implementations and benchmark environments

Employing multi-threading in order to give users more-efficient computing, we (in Rust) implemented ConsTrain and ConsAlign. ConsTrain used the two gamma distribution parameters $\alpha \leftarrow 0, \beta \leftarrow 1$ proposed by Foo *et al* (2009). In ConsTrain, we implemented two different initialization methods for structural alignment scoring parameters to be trained. (1) In the first method, corresponding scoring parameter values were transferred from both the CONTRAfold and the CONTRAlign models. (2) In the other method, structural alignment scoring parameters were initialized by random values derived from the normal distribution with both the mean 0 and the variance $\frac{1}{F}$. ConsAlign used the four lower/upper bounds $i^{\min} \leftarrow 0, i^{\max} \leftarrow 7, j^{\min} \leftarrow 0, j^{\max} \leftarrow 0$ found in Algorithm 2. Also, ConsAlifold performed in ConsAlign took the γ-centroid hyper-parameter $2^0 + 1 = 2$ because this γ-centroid hyper-parameter value showed the highest structure prediction accuracy among the possible γ-centroid hyper-parameter values $\{2^{-4} + 1, \ldots, 2^{10} + 1\}$ in Tagashira and Asai (2022). For our prediction software benchmarks, we used an "Intel Xeon" CPU computer with $240\,\mathrm{GB}$ of RAM, 64 threads, and a clock rate of $2.30\,\mathrm{GHz}$. (We refer to this computer as environment "Xeon.")

3 Results

3.1 Benchmarking ConsAlign to conventional methods

ConsAlign consistently performed the better structure prediction accuracy than other RNA structural aligners, whereas LinearTurboFold (Li *et al*., 2021) has the lowest rank regarding secondary structure prediction accuracy (Fig. 2a–d). ConsAlign showed comparable matching prediction accuracy among state-of-the-art RNA structural aligners, while SPARSE (Will *et al*., 2015) indicated the poorest SPS (Fig. 2e–f). ConsAlign took the median prediction running time among available RNA structural aligners (Table 1).

3.2 Verifying our proposed training and model ensemble

ConsTrain took only about 30 epochs when using our proposed transfer learning (Fig. 3). On the other hand, ConsTrain consumed around 60 epochs when we initialized our alignment scoring parameters by random values (Fig. 3). ConsTrain continuously reduced our L2-regularized non-convex cost in both parameter initialization methods (Fig. 3).

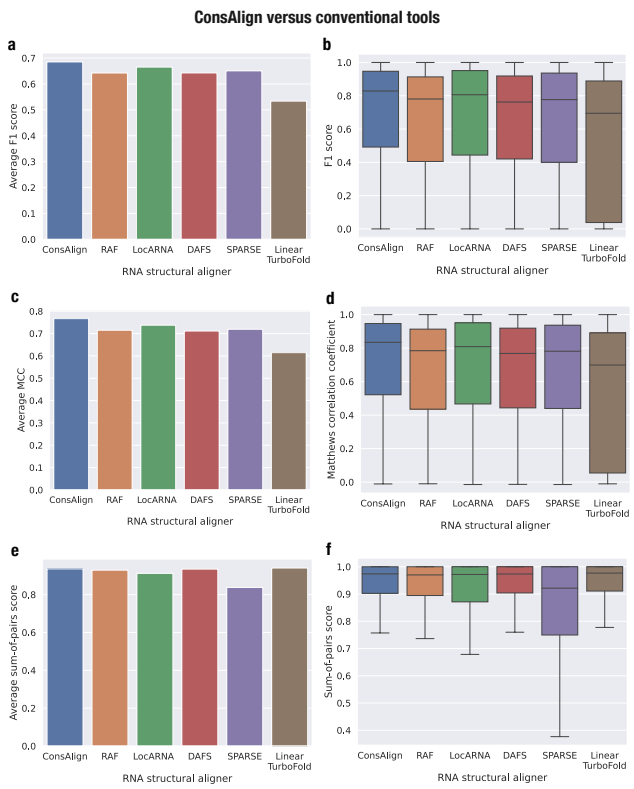

Fig. 2. Alignment prediction accuracy comparison among ConsAlign and conventional structural aligners. Methods were compared in terms of (a) average F1 scores, (c) average MCCs, and (e) average SPSs. Each method's distributions of (b) F1 scores, (d) MCCs, and (f) SPSs are visualized in box plots. Alignment prediction accuracy values taking the denominator 0 were omitted. Outlier SPSs were omitted for SPS box plots. Dataset "test multiple" was used. ConsAlign used our proposed ensemble model with our alignment scoring parameters trained by our proposed transfer learning.

ConsTrain performing our proposed transfer learning demonstrated the best alignment prediction accuracy, especially for F1 scores, among available parameter training schemes (Fig. 4). In addition to old ConsProb alone, our proposed ensemble model exhibited superior structure prediction accuracy compared to new ConsProb alone (Fig. 5a–d). Furthermore, our proposed ensemble model and new ConsProb alone reached higher SPSs than old ConsProb alone (Fig. 5e–f).

4 Conclusion and discussion

We invented ConsTrain—an efficient gradient-based learning method of simultaneous alignment scoring parameters—and applied it to ConsAlign—a model ensemble-based structural aligner. ConsAlign attained better alignment prediction accuracy in our alignment tool benchmarks than existing RNA structural aligners. In our alignment tool benchmarks, LinearTurboFold and SPARSE were the weakest in the prediction accuracy of RNA secondary structure and RNA sequence alignment, respectively. LinearTurboFold is an iterative method different from the other benchmarked simultaneous methods. Therefore, we can interpret that simultaneous solutions enhanced obtained structures' quality by the more direct transmission of structural conservation among homologs than iterative solutions. SPARSE realizes its efficient computation by employing only structure-based constraints, whereas the other simultaneous structural aligners—ConsAlign, LocARNA, RAF, and DAFS—employ matching-based constraints in addition to structure-based constraints. We can explain that leveraging both matching-based and structure-based constraints is more effective regarding obtained nucleotide matches' quality than only structure-based constraints.

ConsTrain with our transfer learning substantiated double parameter training efficiency and better structural alignment quality than our random parameter initialization. ConsTrain's iterative parameter update is not convex due to our attached training hyper-parameters. Thus, we can conclude that providing well-defined scoring parameter values guided ConsTrain to better machine learning solutions. Our proposed ensemble model competed with old ConsProb alone regarding ConsAlign's structure prediction accuracy. Also, our proposed ensemble model compared with new ConsProb alone regarding ConsAlign's matching prediction accuracy.

Table 1. Prediction type and running time of benchmarked structural aligners.

Structural aligner	Structural alignment type	Total running time
ConsAlign	Simultaneous	376 s
RAF		1697 s
LocARNA	⋮	70 s
SPARSE		618 s
DAFS	Simultaneous	84 s
LinearTurboFold	Iterative	1362 s

The center column indicates each aligner's prediction type. The right column shows the total running time of each benchmarked structural aligner on dataset "test multiple." Utilizing `multiprocessing.Pool` (available from `https://docs.python.org/3/library/multiprocessing.html`) to relieve the large volume of dataset "test multiple," each benchmarked structural aligner other than ConsAlign was run on environment "Xeon" by mapping 64 processes. ConsAlign was conducted by providing each of 16 processes with four different threads.

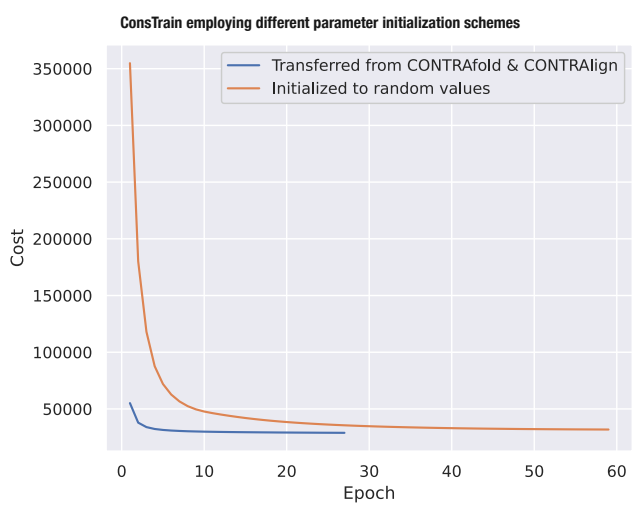

Fig. 3. Decrement of our L2-regularized non-convex cost in different parameter initialization settings. As the first choice, structural alignment scoring parameters were transfer-learned. As the second choice, structural alignment scoring parameters were initialized by random values. Dataset "train pairwise" was used.

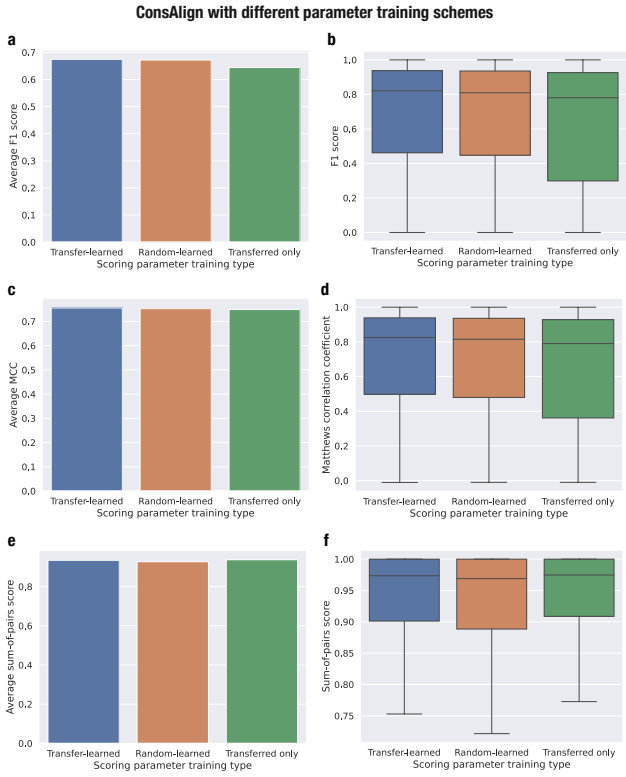

Fig. 4. Alignment prediction accuracy comparison among the different parameter training schemes of ConsTrain. (Transfer-learned) Structural alignment scoring parameters were transfer-learned. (Random-learned) Structural alignment scoring parameters were initialized by random values before training. (Transferred only) Structural alignment scoring parameters were transferred but then were not trained. This figure's plot parameters were the same as Fig. 2. Only new ConsProb (Tagashira and Asai, 2022) parameterized with our transfer learning were used in ConsAlign.

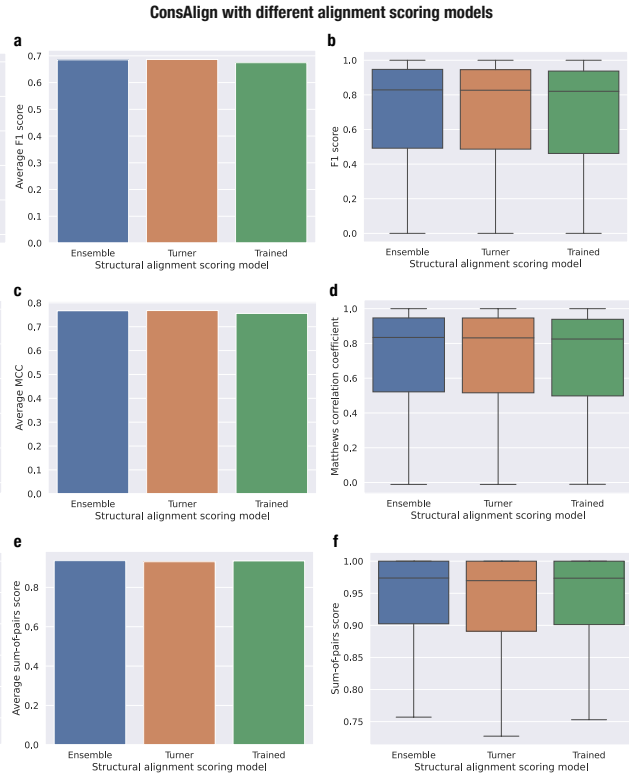

Fig. 5. Alignment prediction accuracy comparison among the different alignment scoring models. (Ensemble) Our proposed ensemble model was used. (Turner) Only old ConsProb was used as a posterior probability inference method. (Trained) Only new ConsProb was used as a posterior probability inference method. This figure's plot configuration is configuration is the same as Fig. 2. Our alignment scoring parameters with our proposed alignment scoring parameters was used in ConsAlign.

Recently, deep learning-based methods for mono-folding—such as SPOT-RNA and MXfold (Singh *et al*., 2019; Sato *et al*., 2021)—emerged replacing non-deep learning methods for mono-folding—such as CONTRAfold and TORNADO (Do *et al*., 2006b; Rivas *et al*., 2012). Expanding ConsTrain into deep learning for simultaneous RNA structural alignments has the potential to improve their quality by deep features unable to capture in any CLLMs. In another scenario, we have a chance to generalize the standard RNA base alphabet (`A`, `C`, `G`, and `U`) used in ConsTrain by including modified bases—such as `I` and Ψ (Harcourt *et al*., 2017). However, we need well-defined training datasets containing modified bases in this scenario because the base composition of training datasets directly affects ConsTrain's trained scoring parameters via gradients. Constructing reliable large structural databases for modified bases can assist machine learning methods involving RNA secondary structures in data-driven avenues.

Acknowledgments

We would like to thank Dr. Kiyoshi Asai, Dr. Martin Frith, and their lab members for a year of discussion. We incredibly appreciate valuable comments from Dr. Kengo Sato and Dr. Akito Taneda regarding points to be improved. We performed most of the computations in this study on the NIG supercomputer at the ROIS National Institute of Genetics, Japan.

Funding

This work was not supported by any funding.

Conflict of Interest

None declared.

Data Availability

Our underlying data are available at our shown GitHub repositories.

References

Aghaeepour,N., Hoos,H.H. (2013) Ensemble-based prediction of RNA secondary structures. *BMC Bioinf.*, **14**, 139.

Andronescu,M. *et al*. (2007) Efficient parameter estimation for RNA secondary structure prediction. *Bioinformatics*, **23**, i19–i28.

Bernhart,S.H. *et al*. (2008) RNAalifold: improved consensus structure prediction for RNA alignments. *BMC Bioinf.*, **9**, 474.

Ding,Y., Chi,Y.C. and Lawrence,C.E. (2005) RNA secondary structure prediction by centroids in a Boltzmann weighted ensemble. *RNA*, **11**, 1157–1166.

Do,C.B., Woods,D.A. and Batzoglou,S. (2006a) CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics*, **22**, e90–e98.

Do,C.B., Gross,S.S. and Batzoglou,S. (2006b) *CONTRAlign: Discriminative Training for Protein Sequence Alignment*. In Proceedings of the Tenth Annual International Conference on Computational Molecular Biology, RECOMB.

Do,C.B., Foo,C. and Batzoglou,S. (2008) A max-margin model for efficient simultaneous alignment and folding of RNA sequences. *Bioinformatics*, **24**, i68–i76.

Dowell,R.D. and Eddy,S.R. (2006) Efficient pairwise RNA structure prediction and alignment using sequence alignment constraints. *BMC Bioinf.*, **7**, 400.

Feng,D. and Doolittle,R.F. (1987) Progressive sequence alignment as a prerequisitetto correct phylogenetic trees. *J. Mol. Evol.*, **25**, 351–360.

Fletcher,R. (1987) *Practical Methods of Optimization*. John Wiley & Sons, Ltd, New York.

Foo,C.S., Do,C.B. and Ng,A.Y. (2009) *A majorization-minimization algorithm for (multiple) hyperparameter learning.* ACM International Conference Proceeding Series.

Hamada,M. *et al*. (2009a) Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics*, **25**, 465–473.

Hamada,M. *et al*. (2009b) Predictions of RNA secondary structure by combining homologous sequence information. *Bioinformatics*, **25**, i330–i338.

Hamada,M. *et al*. (2009c) CentroidAlign: fast and accurate aligner for structured RNAs by maximizing expected sum-of-pairs score.

Hamada,M., Sato,K. and Asai,K. (2010) Prediction of RNA secondary structure by maximizing pseudo-expected accuracy. *BMC Bioinf.*, **11**, 586. *Bioinformatics*, **25**, 3236–3243.

Hamada,M., Sato,K. and Asai,K. (2011) Improving the accuracy of predicting secondary structure for aligned RNA sequences. *Nucleic Acids Res.*, **39**, 393–402.

Harcourt,E.M., Kietrys,A.M. and Kool,E.T. (2017) Chemical and structural effects of base modifications in messenger RNA. *Nature*, **541**, 339–346.

Kalvari,I. *et al*. (2018) Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families. *Nucleic Acids Res.*, **46**, D335–D342.

Knudsen,B. and Hein,J. (2003) Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Res.*, **31**, 3423–3428.

Lafferty,J., McCallum,A. and Pereira,F. (2001) *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.* Proceedings of the 18th International Conference on Machine Learning 2001, 282–289.

Langed,K., Hunter,A.R. and Yang,I. (2000) Optimization Transfer Using Surrogate Objective Functions. *J. Comput. Graph. Stat.*, **9**, 1–20.

Li,S. *et al*. (2021) LinearTurboFold: Linear-time global prediction of conserved structures for RNA homologs with applications to SARS-CoV-2. *Proc. Natl. Acad. Sci. U. S. A.*, **118**.

Rivas,E., Lang,R. and Eddy,S.R. (2012) A range of complex probabilistic models for RNA secondary structure prediction that includes the nearest-neighbor model and more. *RNA*, **18**, 193–212.

Sankoff,D. (1985) Simultaneous Solution of the RNA Folding, Alignment and Protosequence Problems. *SIAM J. Appl. Math.*, **45**, 810–825.

Sato,K. and Sakakibara,Y. (2005) RNA secondary structural alignment with conditional random fields. *Bioinformatics*, **21**, ii237–ii242.

Sato,K. *et al*. (2012) DAFS: simultaneous aligning and folding of RNA sequences via dual decomposition. *Bioinformatics*, **28**, 3218–3224.

Sato,K., Akiyama,M. and Sakakibara,Y. (2021) RNA secondary structure prediction using deep learning with thermodynamic integration. *Nat Commun.*, **12**, 1–9.

Sato,K. and Kato,Y. (2022) Prediction of RNA secondary structure including pseudoknots for long sequences. *Brief Bioinform.*, **23**, 1–9.

Seemann,S.E., Gorodkin,J. and Backofen,R. (2008) Unifying evolutionary and thermodynamic information for RNA folding of multiple alignments. *Nucleic Acids Res.*, **36**, 6355–6362.

Singh,J. *et al*. (2019) RNA secondary structure prediction using an ensemble of two-dimensional deep neural networks and transfer learning. *Nat. Commun.*, **10**, 1–13.

Sneath,P.H. and Sokal,R.R. (1962) Numerical taxonomy. *Nature*, **193**, 855–860.

Tagashira,M. and Asai,K. (2022) ConsAlifold: considering RNA structural alignments improves prediction accuracy of RNA consensus secondary structures. *Bioinformatics*, **38**, 710–719.

Tan,Z. *et al*. (2017) TurboFold II: RNA structural alignment and secondary structure prediction informed by multiple homologs. *Nucleic Acids Res.*, **45**, 11570–11581.

Turner,D.H. and Mathews,D.H. (2010) NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Res.*, **38**, D280–D282.

Will,S., Reiche,K., Hofacker,I.L., Stadler,P.F., Backofen,R. (2007) Inferring noncoding RNA families and classes by means of genome-scale structure-based clustering. *PLoS Comput. Biol.*, **3**, 680–691.

Will,S. *et al*. (2015) SPARSE: quadratic time simultaneous alignment and folding of RNAs without sequence-based heuristics. *Bioinformatics*, **31**, 2489–2496.

Zakov,S., Goldberg,Y., Elhadad,M. and Ziv-Ukelson, M. (2011) Rich parameterization improves RNA structure prediction. *J. Comput. Biol.*, **18**, 1525–1542.

Zhang,H., Zhang,L., Mathews,D.H. and Huang,L. (2020) LinearPartition: Linear-time approximation of RNA folding partition function and base-pairing probabilities. *Bioinformatics*, **36**, i258–i267.