RNA Contact Prediction by Data Efficient Deep Learning (Supplementary Information)

(Dated: December 2, 2022)

I. DATASETS

Rfam 14.6 [1, 2] is a database of RNA families, each comprising a multiple sequence alignment, consensus secondary structure and a covariance model. Seed alignments of Rfam dataset contain manually curated set of the most representative sequences of the family, whilst the full ones comprise all known members. For each available MSA in Rfam we use full alignment only in case the corresponding seed alignment consists of less than 100 sequences. Overall there are 4070 MSAs each comprising between 10 and 5000 sequences.

Zasha Weinberg Data [3] are the alignments available on `https://bitbucket.org/zashaw/zashaweinbergdata/src/master/`. We use 43 MSAs which were not previously submitted to Rfam database with numbers of sequences ranging between 8 and 650 in each. We combine these as our pre-training dataset. For cross-validation we hold back a random set of MSAs from the combined set.

The dataset used in [4] is based on the one presented in [5]. The latter one was curated by increasing E-value cutoff of the Rfam 14.1 RNA families to 0.99 and using the cmsearch option of Infernal [6] without modifying covariance models. Out of 70 RNA structures with high resolution 57 were selected thereafter and used as a training set. Additional 23 RNA structures were collected by relaxing the structure resolution criteria of [5] and used as a test set. Number of sequences in the MSAs ranges between 6 and 190000.

TABLE I. Pre-training and finetuning parameters of the deep-learning-based models.

Parameter	Pre-training	Finetuning
Architecture		
# Attention blocks	10	10
# Heads per block	12	12
# Features per head	64	64
Batch size (local)	1	1
Optimizer	Adam	Adam
Learning rate	3×10^{-5}	1×10^{-4}
Warm-up	linear, 400 epochs	–
Decay	inverse square root, after 400 epochs	–
Drop-out ratio	0.3	0
Pre-processing		
Cropping mode	random	random
Cropping size	400	400
Subsampling mode	random	diversity-maximizing
Subsampling size	50	50
Inpainting		
Mode	token-wise	–
Masking ratio	0.15	–
Replacing tokens	regular seq. tokens	–
Jigsaw		
# Chunks	4	–
# Permutations	24	–
Bootstrapping		
Mode	token-wise	–
Replacement ratio	0.5	–
Contrastive		
Temperature	100	–
Validation data size	100 MSAs	20%
Float precision	16-bit mixed	16-bit mixed
Training time	2 days	$<$ 30 minutes (incl. early stopping)

II. TRAINING DETAILS

A. Loss Functions

In general, all our upstream and the downstream task are modeled as classification problems. We therefore use different variations of cross-entropy loss to train our models: Inpainting and Jigsaw employ categorical cross-entropy, Bootstrapping uses binary cross-entropy, and normalized temperature-scaled cross-entropy loss (NT-Xent loss, [7]) is applied to the Contrastive task. In the downstream training of the regression model, we use focal loss [8] with parameters $\alpha = 0.95, \gamma = 2$ to counteract the class imbalance between contacts and non-contacts.

In case of a conjunction of several upstream tasks, we define the total loss $\mathcal{L}$ used for training as unweighted sum of the individual task-specific losses

$$\mathcal{L} = \sum_{t \in \mathcal{T}} \mathcal{L}_t, \tag{1}$$

where $\mathcal{T} \subseteq \{\text{inpainting}, \text{jigsaw}, \text{contrastive}, \text{bootstrapping}\}$ denotes the set of tasks being combined in the training of a specific upstream model and $\mathcal{L}_{\text{inpainting}}$, $\mathcal{L}_{\text{jigsaw}}$, $\mathcal{L}_{\text{bootstrapping}}$, and $\mathcal{L}_{\text{contrastive}}$ denote the corresponding individual task losses. We also tested a weighted sum, but found that these additional parameters did not actually improve our results.

B. Training Parameters

Table I summarizes the parametrization of our deep-learning-based models.

Pre-training Parameters

The upstream model consists of 10 attention blocks with 12 heads and 64 features per head, totalling almost 60M parameters. We use a local batch size of 1 and train each upstream model for 2 days. Adam is em-

TABLE II. Training parameters of the XGBoost models. Other parameters are set to their default values.

Parameter	Value
# Trees (max)	300
Tree depth (max)	16
Learning rate	1.0
Booster	DART [9]
Drop-out ratio	0.1
Subsampling	
Mode	gradient-based
Rate	0.9
Colsample	
By-tree	0.7
By-level	0.7
Minimum split loss	0.7
Objective	binary:logitraw
Tree method	gpu_hist
Training time	$<$ 10 minutes (incl. early stopping)

ployed as optimization algorithm. The learning rate is set to 3×10^{-5} with an linear warm-up phase of 400 epochs and a subsequent decay according to an inverse square root rule. To increase the training speed, we use 16-bit mixed precision. Furthermore, the dropout layers in the attention blocks operate with a drop-out ratio of 0.3. In order to stay within the memory limits of our system, we crop and subsample large MSAs randomly to sequence length 400 and sequence count 50, respectively. Performance metrics (e.g., accuracy) are evaluated on a random subset of 100 MSAs excluded from the training dataset.

Inpainting is performed token-wise, with a masking ratio of 0.15. Masked tokens are replaced by a random token from the set of regular sequence tokens. For Jigsaw, we divide sequences into 4 chunks and allow all 24 permutations. Similar to Inpainting, the token-wise implementation of the Bootstrapping task has also proven to be best. Here, however, we use a replacement ratio of 0.5. Other values did not lead to significant improvement in the upstream performance. Last, the temperature in the Contrastive task is set to 100.

Downstream Training Parameters

While architectural parameters of the backbone remain the same, we adapt some of the training parameters compared to the pre-training. For example, since the labeled training dataset is drastically smaller than the unlabeled dataset used for pre-training, we are able to use diversity-maximizing subsampling as in [10]. Moreover, a random selection of 20% of the training data samples is withheld from training and used as a validation dataset.

The regression model is trained for at most 30 minutes, but we early stop w.r.t. the performance metrics top-L-precision, all-positive top-L-precision, F1 score, and Matthews correlation coefficient on the validation data. Both dropout and learning rate scheduling are disabled, and the learning rate is set to 1×10^{-4}. These settings apply to both the training of the actual regression model and the re-training of the backbone for the finetuned XGBoost model.

Table II contains the hyperparameters of the XGBoost models. Every XGBoost model consists of at most 300 trees with maximum depth 16. It is built using gradient-based subsampling, where the subsampling rate is set to 0.9. Again, we perform early stopping w.r.t. to the above metrics on the validation data. We also employ a dropout-like extension called DART [9] with drop-out rate 0.1. Aside from that, we use the following parametrization: learning-rate 1.0, minimum-split-loss 0.7, minimum-child-weight 1.0, colsample-bytree 0.7, colsample-bylevel 0.7.

C. Parallelization

We employ data-parallelism to further speed-up our pre-training by distributing the load to four GPUs. The downstream training, however, is conducted only on one GPU. For the XGBoost model, this is because its fitting cannot be data-parallelized. For the regression model, on the other hand, we do not run into training time issues as for pre-training, so we rather aim to keep the effective batch size minimal in order to achieve the maximum effect of stochastic gradient descent.

The parallelization strategy used for pre-training works as follows: First, before the actual training is started, the training dataset is distributed to the allocated GPUs, i.e., each GPU obtains visibility only into a subset of the overall dataset. During training, gradient steps are synchronized: All GPUs perform a forward and backward pass and subsequently wait for each other. Then, the computed local gradients are collected, shared and averaged. In the end, weights on each GPU are updated by the same global gradient.

III. HYPERPARAMETER SEARCH AND ABLATION STUDIES

We tune the hyperparameters for the upstream and the downstream separately. For the backbone we use a random search in the limits in table III. We run 300 iterations of this search and arrive at the parameters given in table I.

For the downstream regression model we only experimented with adding a hidden layer and adding biases, none of which showed any benefit.

We use an evolutionary optimization[11] to explore the parameter space for the XGBoost models. Parameter limits are listed in table IV. We use 16 workers to evaluate individual candidate solutions in parallel for about 6000 generations. The population size is set to 16 as

TABLE III. Parameter limits for the optimization of hyperparameters in the upstream training.

Parameter	Limits
# blocks	{6, 8, 10}
# heads	{8, 12, 16}
d_{head}	{16, 32, 64, 128}
learning rate	$[10^{-6}, 10^{3}]$
dropout	$(0, 50]$
Inpainting maskin mode	{token, column}
Jigsaw partitions	{3, 4, 5}
Contrastive Temperature	{10, 100}

TABLE IV. Parameter limits for the optimization of hyperparameters in the XGBoost training.

Parameter	Limits
# Trees (max)	$[1, 500]$
Tree depth (max)	$[4, 16]$
Learning rate	$[0.01, 1.0]$
Drop-out ratio	$[0, 0.5]$
Subsampling	
Mode	{uniform, gradient-based}
Rate	$[0.4, 1]$
Colsample	
By-tree	$[0.4, 1]$
By-level	$[0.4, 1]$
Minimum split loss	$[0, 1]$

well, i.e., each worker operates on a single parametrization candidate. The algorithm to generate new individual candidates is a simple combination of cross-breeding, point mutation, and selection for top-L precision. Moreover, each of the 4 disjointed sets of 4 workers forms an island being isolated from the others with respect to cross-breeding. The probability for migration between those islands is chosen 0.1. Since a single XGBoost training only runs for few minutes instead of days, the energy footprint of this search is negligible next to the unsupervised backbone training.

IV. INTERPRETATION OF ATTENTION MAPS

Figure 1 illustrated the relative importance for the downstream input feature maps originating from individual attention heads. The regression model using a frozen backbone focuses more on specific and on earlier heads. The finetuning spread the focus of the model over evenly over mostly deeper heads. The XGBoost model focuses already on later heads without finetuning. Using features from the finetuned backbone as input for the XGBoost model focuses it on heads of intermediate depth, but does not make it as diffuse as the finetuned regression.

V. EXTENDED RESULTS TABLE

Tables V, VI and VII show all measured metrics to an extended set of models, that were left out of the main paper for clarity and brevity.

VI. ADDITIONAL EXAMPLE

Figures 2 and 3 show an additional example to the one in the main text. The cluster of false positive predictions seems to originate from the presence of another molecule in the reference of 5di2. If that molecule is not present the gap might be closed as predicted by our model. To deal with this situation consistently, either the dataset should include strictly monomeric structures, or multimeric structures would have to explicitely included and represented sufficiently in the dataset so the model has an opportunity to learn these intricacies.

[1] I. Kalvari, E. P. Nawrocki, J. Argasinska, N. Quinones-Olvera, R. D. Finn, A. Bateman, and A. I. Petrov, Non-coding RNA analysis using the Rfam database, Current protocols in bioinformatics **62**, e51 (2018).

[2] I. Kalvari, E. P. Nawrocki, N. Ontiveros-Palacios, J. Argasinska, K. Lamkiewicz, M. Marz, S. Griffiths-Jones, C. Toffano-Nioche, D. Gautheret, Z. Weinberg, *et al.*, Rfam 14: expanded coverage of metagenomic, viral and microRNA families, Nucleic Acids Research **49**, D192 (2021).

[3] Z. Weinberg, C. E. Lünse, K. A. Corbino, T. D. Ames, J. W. Nelson, A. Roth, K. R. Perkins, M. E. Sherlock, and R. R. Breaker, Detection of 224 candidate structured RNAs by comparative analysis of specific subsets of intergenic regions, Nucleic acids research **45**, 10811 (2017).

[4] M. B. Zerihun, F. Pucci, and A. Schug, CoCoNet—boosting RNA contact prediction by convolutional neural networks, Nucleic acids research **49**, 12661 (2021).

[5] F. Pucci, M. B. Zerihun, E. K. Peter, and A. Schug, Evaluating DCA-based method performances for RNA contact prediction by a well-curated data set, RNA **26**, 794 (2020).

[6] E. P. Nawrocki and S. R. Eddy, Infernal 1.1: 100-fold faster RNA homology searches, Bioinformatics **29**, 2933 (2013).

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, A simple framework for contrastive learning of visual representations, in *Proceedings of the 37th International Conference on Machine Learning* (PMLR, 2020) pp. 1597–1607.

[8] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, Focal loss for dense object detection, in *2017 IEEE International Conference on Computer Vision (ICCV)* (2017) pp. 2999–3007.

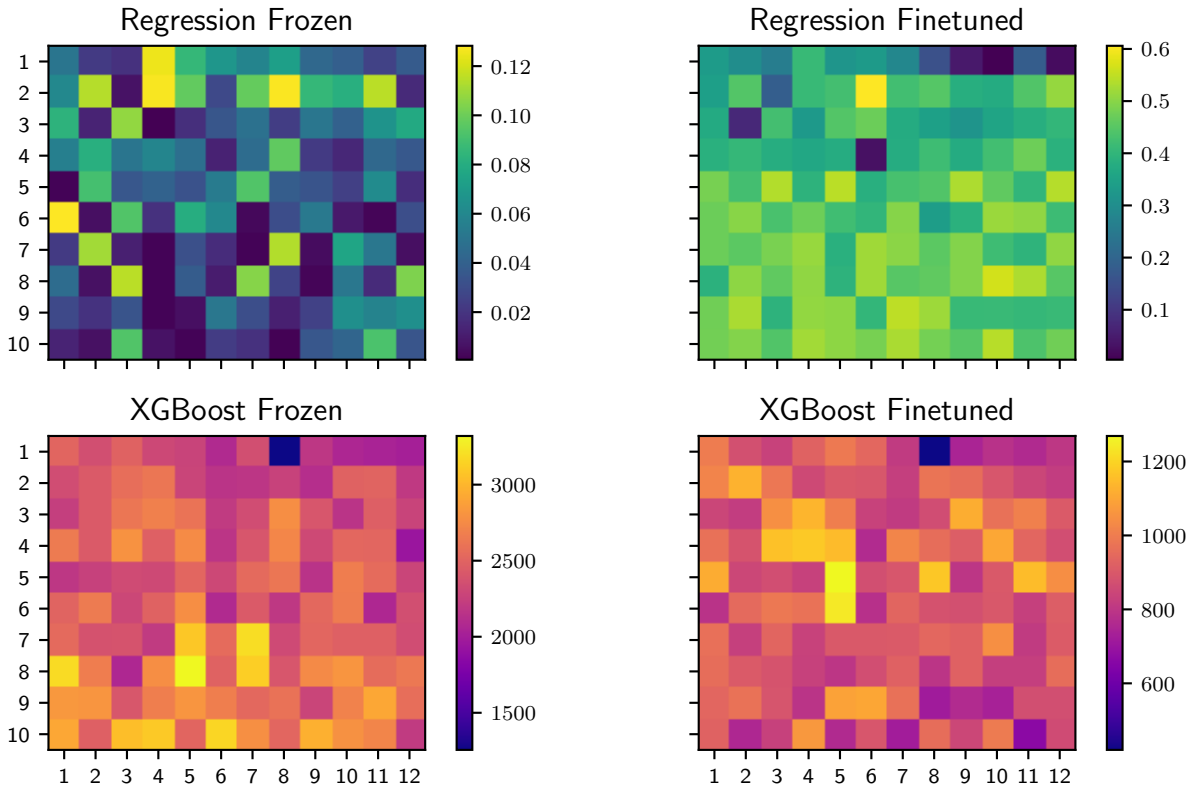

FIG. 1. Absolute values of the regression model parameters (top) and XGBoost feature importance scores (bottom) for a selection of downstream models with fixed (left) and finetuned (right) backbone, respectively. The regression models are pre-trained with Bootstrapping and optimized for global F1 score. The XGBoost models are pre-trained with just Inpainting and optimized for top-L precision.

[9] R. K. Vinayak and R. Gilad-Bachrach, DART: Dropouts meet Multiple Additive Regression Trees, in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics* (PMLR, 2015) pp. 489–497.

[10] R. M. Rao, J. Liu, R. Verkuil, J. Meier, J. Canny, P. Abbeel, T. Sercu, and A. Rives, MSA Transformer, in *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 139 (PMLR, 2021) pp. 8844–8856.

[11] Will be provided upon publication.

Task + checkpoint metric(s)	$PPV_{L;0.5}$/%	PPV_L / %	PPV / %	SEN/%	$F1$/%	MCC/%	Energy / Wh
frozen regression							
inpainting F1	47.33	47.33	9.05	97.62	16.57	3.59	**3.57**
inpainting MCC	36.16	36.16	**10.31**	59.92	**17.59**	**5.37**	4.74
jigsaw F1	23.57	23.57	8.82	**100.00**	16.22	0.16	3.59
jigsaw MCC	23.70	23.70	8.83	**99.98**	16.23	0.81	3.76
contrastive F1	46.31	46.31	8.93	98.55	16.38	2.36	3.65
contrastive MCC	15.16	15.16	8.68	50.96	14.84	-0.51	3.70
bootstrap F1	**53.76**	**53.76**	8.95	**99.12**	16.42	2.92	4.06
bootstrap MCC	26.97	26.97	**10.15**	42.32	16.37	3.56	4.08
finetuned regression							
inpainting $PPV_{L;0.5}$	**84.33**	**84.33**	19.23	72.02	30.36	25.78	4.54
inpainting F1	75.85	75.85	23.08	73.21	35.09	31.33	4.56
jigsaw $PPV_{L;0.5}$	67.31	67.31	15.50	**77.33**	25.83	20.88	4.93
jigsaw F1	60.76	60.76	19.34	66.85	30.00	24.56	4.88
contrastive $PPV_{L;0.5}$	**84.01**	**84.01**	18.84	71.18	29.79	24.97	3.68
contrastive F1	67.50	67.50	26.35	67.90	37.96	33.52	**3.65**
bootstrap $PPV_{L;0.5}$	81.89	81.89	18.52	71.38	29.41	24.54	3.77
bootstrap F1	69.49	69.49	**30.49**	61.95	**40.87**	**35.71**	3.74
frozen XGBoost							
inpainting $PPV_{L;0.5}$	83.72	79.90	78.71	24.08	36.88	41.04	3.90
inpainting F1	70.46	70.46	53.77	28.62	37.35	35.18	**2.59**
jigsaw $PPV_{L;0.5}$	**87.08**	43.93	**87.08**	4.78	9.07	19.26	3.81
jigsaw F1	26.01	26.01	18.35	16.90	17.60	9.99	3.63
contrastive $PPV_{L;0.5}$	85.46	**83.17**	77.26	26.53	**39.50**	**42.65**	4.12
contrastive F1	71.16	71.16	50.70	**31.16**	**38.60**	35.36	2.77
bootstrap $PPV_{L;0.5}$	**87.07**	81.44	79.90	23.68	36.53	41.06	3.78
bootstrap F1	73.43	73.09	56.12	28.54	37.84	36.14	3.17
finetuned XGBoost							
inpainting $PPV_{L;0.5}$ $PPV_{L;0.5}$	**86.06**	**86.06**	**61.40**	46.63	**53.00**	**49.68**	3.57
inpainting MCC $PPV_{L;0.5}$	76.24	76.24	50.28	**53.80**	51.98	47.20	2.81
inpainting $PPV_{L;0.5}$ MCC	**85.36**	**85.36**	59.18	46.95	52.36	48.71	3.65
inpainting MCC MCC	78.93	78.93	52.81	52.82	52.81	48.25	3.73
jigsaw $PPV_{L;0.5}$ $PPV_{L;0.5}$	68.46	68.46	46.20	41.47	43.71	38.65	5.44
jigsaw MCC $PPV_{L;0.5}$	63.33	63.33	41.27	41.55	41.41	35.72	4.07
jigsaw $PPV_{L;0.5}$ MCC	68.91	68.91	45.90	41.58	43.63	38.53	2.75
jigsaw MCC MCC	62.68	62.68	40.46	41.78	41.11	35.32	3.82
contrastive $PPV_{L;0.5}$ $PPV_{L;0.5}$	84.14	84.14	58.19	44.74	50.59	46.96	3.63
contrastive MCC $PPV_{L;0.5}$	75.59	75.59	47.49	52.51	49.88	44.83	2.97
contrastive $PPV_{L;0.5}$ MCC	84.14	84.14	58.19	44.74	50.59	46.96	3.04
contrastive MCC MCC	75.72	75.72	47.45	52.49	49.85	44.79	**2.61**
bootstrap $PPV_{L;0.5}$ $PPV_{L;0.5}$	84.01	84.01	59.26	43.42	50.12	46.75	3.71
bootstrap MCC $PPV_{L;0.5}$	74.69	74.69	49.47	48.51	48.99	44.11	3.73
bootstrap $PPV_{L;0.5}$ MCC	84.01	84.01	59.26	43.42	50.12	46.75	3.65
bootstrap MCC MCC	74.69	74.69	49.47	48.51	48.99	44.11	3.32

TABLE V. Downstream contact prediction metrics for a selection of models. The metrics displayed in the left most column are the monitor metrics used for early stopping. The energy measurement only includes the energy for the downstream training itself, not the upstream training. PPV_L is the precision computed over the top L predicted scores in one sample and then averaged over all samples, assuming the model made at least L positive predictions per sample. For models that made fewer than L predictions this would effectively mean lowering the decision boundary. $\mathrm{PPV}_{L;0.5}$ is the precision computed over the top L predictions with unmoved decision threshold. PPV, SEN, F1, and MCC are computed over all scores of a sample and then averaged over all samples. Values within a single percentage point of the best result are highlighted.

Task + checkpoint metric(s)	$PPV_{L;0.5}$/%	PPV_L / %	PPV / %	SEN/%	$F1$/%	MCC/%	Energy / Wh
frozen XGBoost							
inpainting $PPV_{L;0.5}$	83.72	79.90	78.71	24.08	36.88	41.04	3.90
inpainting F1	70.46	70.46	53.77	28.62	37.35	35.18	**2.59**
jigsaw $PPV_{L;0.5}$	**87.08**	43.93	**87.08**	4.78	9.07	19.26	3.81
jigsaw F1	26.01	26.01	18.35	16.90	17.60	9.99	3.63
contrastive $PPV_{L;0.5}$	85.46	**83.17**	77.26	26.53	**39.50**	**42.65**	4.12
contrastive F1	71.16	71.16	50.70	**31.16**	**38.60**	35.36	2.77
bootstrap $PPV_{L;0.5}$	**87.07**	81.44	79.90	23.68	36.53	41.06	3.78
bootstrap F1	73.43	73.09	56.12	28.54	37.84	36.14	3.17

TABLE VI. Downstream contact prediction metrics for a selection of models. The metrics displayed in the left most column are the monitor metrics used for early stopping. The energy measurement only includes the energy for the downstream training itself, not the upstream training. PPV_L is the precision computed over the top L predicted scores in one sample and then averaged over all samples, assuming the model made at least L positive predictions per sample. For models that made fewer than L predictions this would effectively mean lowering the decision boundary. $\mathrm{PPV}_{L;0.5}$ is the precision computed over the top L predictions with unmoved decision threshold. PPV, SEN, F1, and MCC are computed over all scores in one sample and then averaged over all samples. Results within a single percentage point of the best result are highlighted.

Task + checkpoint metric(s)	$PPV_{L;0.5}$/%	PPV_L / %	PPV / %	SEN/%	$F1$/%	MCC/%	Energy / Wh
finetuned XGBoost							
inpainting $PPV_{L;0.5}$ $PPV_{L;0.5}$	**86.06**	**86.06**	**61.40**	46.63	**53.00**	**49.68**	3.57
inpainting MCC $PPV_{L;0.5}$	76.24	76.24	50.28	**53.80**	51.98	47.20	2.81
inpainting $PPV_{L;0.5}$ MCC	**85.36**	**85.36**	59.18	46.95	52.36	48.71	3.65
inpainting MCC MCC	78.93	78.93	52.81	52.82	52.81	48.25	3.73
jigsaw $PPV_{L;0.5}$ $PPV_{L;0.5}$	68.46	68.46	46.20	41.47	43.71	38.65	5.44
jigsaw MCC $PPV_{L;0.5}$	63.33	63.33	41.27	41.55	41.41	35.72	4.07
jigsaw $PPV_{L;0.5}$ MCC	68.91	68.91	45.90	41.58	43.63	38.53	2.75
jigsaw MCC MCC	62.68	62.68	40.46	41.78	41.11	35.32	3.82
contrastive $PPV_{L;0.5}$ $PPV_{L;0.5}$	84.14	84.14	58.19	44.74	50.59	46.96	3.63
contrastive MCC $PPV_{L;0.5}$	75.59	75.59	47.49	52.51	49.88	44.83	2.97
contrastive $PPV_{L;0.5}$ MCC	84.14	84.14	58.19	44.74	50.59	46.96	3.04
contrastive MCC MCC	75.72	75.72	47.45	52.49	49.85	44.79	**2.61**
bootstrap $PPV_{L;0.5}$ $PPV_{L;0.5}$	84.01	84.01	59.26	43.42	50.12	46.75	3.71
bootstrap MCC $PPV_{L;0.5}$	74.69	74.69	49.47	48.51	48.99	44.11	3.73
bootstrap $PPV_{L;0.5}$ MCC	84.01	84.01	59.26	43.42	50.12	46.75	3.65
bootstrap MCC MCC	74.69	74.69	49.47	48.51	48.99	44.11	3.32

TABLE VII. Downstream contact prediction metrics for a selection of models. The metrics displayed in the left most column are the monitor metrics used for early stopping. The first metric is the checkpoint used for the backbone finetuning and the second one is the monitor metric used for the XGBoost training itself. The energy measurement only includes the energy for the downstream training itself, not the upstream training. PPV_L is the precision computed over the top L predicted scores in one sample and then averaged over all samples, assuming the model made at least L positive predictions per sample. For models that made fewer than L predictions this would effectively mean lowering the decision boundary. $\mathrm{PPV}_{L;0.5}$ is the precision computed over the top L predictions with unmoved decision threshold. PPV, SEN, F1, and MCC are computed over all scores and then averaged over all samples. Results within a single percentage point of the best result are highlighted.

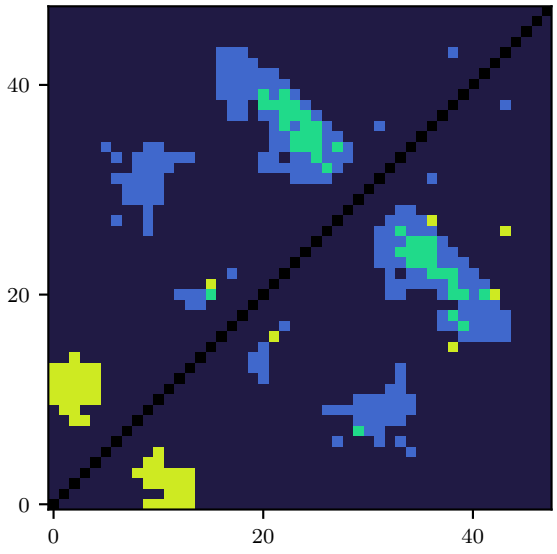

FIG. 2. Contact map (PDB: 5di2) - unfrozen vs. frozen. The upper left part shows the top L contact predictions for the best model with unfrozen backbone, the lower right one for the best model with frozen backbone. Green pixels refer to true positives, yellow to false positives, light blue to false negatives, and dark blue to true negatives.

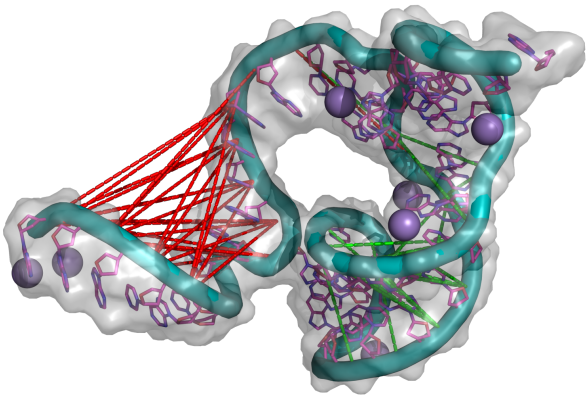

FIG. 3. 3D visualization of an RNA (PDB: 5di2). Green dashed lines indicate correctly predicted inter-residue contacts, red ones refer to false positives.