AutoRNA: RNA tertiary structure prediction using variational autoencoder.

M.A. Kazanskii[1], L. Uroshlev[1], F. Zatylkin[1], I. Pospelova[1], O.Kantidze[1], Y.Gankin[1]

[1] Quantori LLC, Cambridge, MA, USA

*Corresponding author: mkazanskii@gmail.com

Abstract

The understanding of RNA tertiary structure is essential for advancing therapeutic development and vaccine design. Traditional methods, such as dynamic simulations, have been employed to study RNA structure but often struggle to capture the complex, non-linear relationships within RNA sequences. Many previous approaches relied on simpler models, which limited their performance. A key challenge in predicting RNA tertiary structure is the scarcity of available data. To address this, we introduce a Variational Autoencoder (VAE), which achieves a Root Mean Square Error (RMSE) of approximately 4.5 Å in predicting nucleotide positions.

Introduction

To develop new RNA-based medications, it is crucial to understand the fundamental properties of various RNA molecules [1], including their three-dimensional (3d) structure. Leveraging machine learning (ML) techniques to decipher protein folding can pave the way for resolving comparable challenges in the RNA domain. AlphaFold2 [2] showed the power of ML methods for solving protein folding problems. Nevertheless, adapting neural networks similar to AlphaFold2 for RNA could present challenges. The RNA folding problem might seem less complex due to a smaller variety of nucleotides compared to the amino acids in proteins, which reduces the number of viable options. However, the datasets of RNA structures available for training are significantly smaller than those for proteins [3], which may hinder the effectiveness of alignment-based ML approaches [2] when applied to RNA. As of this writing, about 4500 structures in the PDB contain RNA and over 200000 protein structures. Therefore, the applicability of machine learning methods based on the coevolution of monomers, like AlphaFold2[2], for RNA cases may be limited. Studies have also indicated that the energy landscape of 3D RNA structures could be more intricate than that of proteins [3], with RNA potentially exhibiting a more substantial number of local energy minima. Furthermore, understanding of nucleotide coevolution in RNA is limited, mainly because RNA encompasses distinct classes such as tRNA, miRNA, and siRNA, each with its own evolutionary path. Moreover, the way in which RNA evolved for prokaryotes, eukaryotes, and viruses were quite different due to the different mechanisms of regulation and translation.

As a result, for predicting 3D RNA structures, it is necessary to employ ML techniques that do not solely rely on alignment or empirical knowledge of nucleotide coevolution. Precise determination of 3D structure requires ML methods that either do not use alignment and coevolutionary data or utilize such information with a high degree of caution.

Here, we present a novel simple architecture for predicting the 3D structure of RNA sequences. Our approach is based on the variational autoencoder. Since the dataset is small, a simple generative model is justified. We were able to reach an average RMSE of approximately 4.5Å.

Related Work

Numerous algorithms have been developed for predicting the 3D structure of RNA, although the accuracy of these predictions is not yet exceptionally high. Starting from the problem of predicting the secondary structure of RNA, several attempts have been made to generalize methods and algorithms for the 3D case. In [4], the authors tried to apply methods of dynamic programming for 3D structure prediction. The most utilized techniques are based on the empirical potential method and its variants, which calculate the interaction forces between RNA atoms by examining interatomic contacts [6,7]. In these methods, the interaction forces between RNA atoms are determined from the analysis of interatomic contacts. This approach is also used for protein or protein-ligand interaction modeling [8]. Notable empirical potentials include AMBER [9] and CHARMM [10], which are integrated into widely used molecular dynamics software. However, employing molecular dynamics directly for modeling is a time-consuming process. The complete folding of an RNA molecule requires 10^{-6} seconds for forming the 3D backbone structure and 10^{-4} seconds for folding each nucleotide in the molecule [11]. Therefore, the folding time could be 0.01-0.1 second for a molecule of several tens of nucleotides, so direct folding of such structures could be expensive. Coarse-grained potentials are one of the possible ways [12] to reduce complexity and speed up computations.

Recently, hybrid methods have become more popular. These methods combine empirical potentials with various neural network structures. For instance, one study has applied the transformer architecture in conjunction with Rosetta's empirical potentials [12], while another has merged coarse-grained empirical potential with Generative Adversarial Networks (GANs) [13].

An alternative approach derived from protein design is the assembly of structures using elementary templates. A technique described in one of the studies involves breaking down structures into a collection of 3D templates and then optimizing these through a minimization process in the final phase of the algorithm. Reference [14] describes a web server that decomposes structures into a set of 3D templates, followed by a minimization process in the final stage of the algorithm.

In addition, various kinds of ML approaches are used to predict the structure. In [15], a modified UNet neural network predicts a distance map for 3D RNA structure. A promising approach was shown in [18] where structure sampling of FARFAR2 and Rosetta is combined with deep learning scores.

Methodology

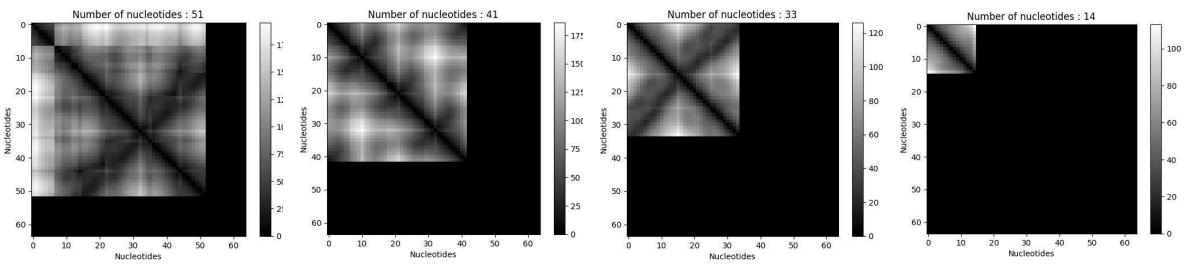
Data preprocessing and experiment

We used the Protein Data Bank (PDB) dataset, which contains 3D structural information of the RNA molecules [3]. We choose PDB files that contain only RNA structure without complexes with protein or DNA.

The data processing involved several steps: we removed the duplicates and the empty sequences, dropped molecules from the chain, and calculated the spatial positions of the backbone of each of the nucleotides. We used the backbone coordinates as a tertiary structure for the RNA sequence. For the main experiment, we selected only those RNA sequences that were 64 nucleotides or shorter. The distance matrix was generated to represent the distances between each nucleotide backbone. For the nucleotide sequences with a number of nucleotides (or sequence length) less than 64, we used padding with a value of 0. Additionally, we clipped the values of the distance. Indeed, the distance mostly lay in the interval (0 Å, 80 Å), and only 0.06% of the distances exceeded the value of 100 Å. Therefore, fixing the values to 100 for the distances exceeding 100 would not significantly affect the algorithm's performance. The examples of the distance matrices are shown in Figure 1. The distance matrices were normalized in the range of [0,1] for the input to the autoencoder. The values of the normalized distance matrices were determined as:

$$d\,m_{norm} = \frac{dm}{100}$$

Figure 1. Examples of the input images to the ML pipeline. Each image is equivalent to the distance matrix corresponding to the RNA sequence with zero padding. The images were scaled by a factor of 255 for better visualization: original images have pixel values in range (0,1). On the X and Y axes, the nucleotides are represented. For the sake of sanity, we have not included the sequences. The black area corresponds to the padding (to reach 64 nucleotides).

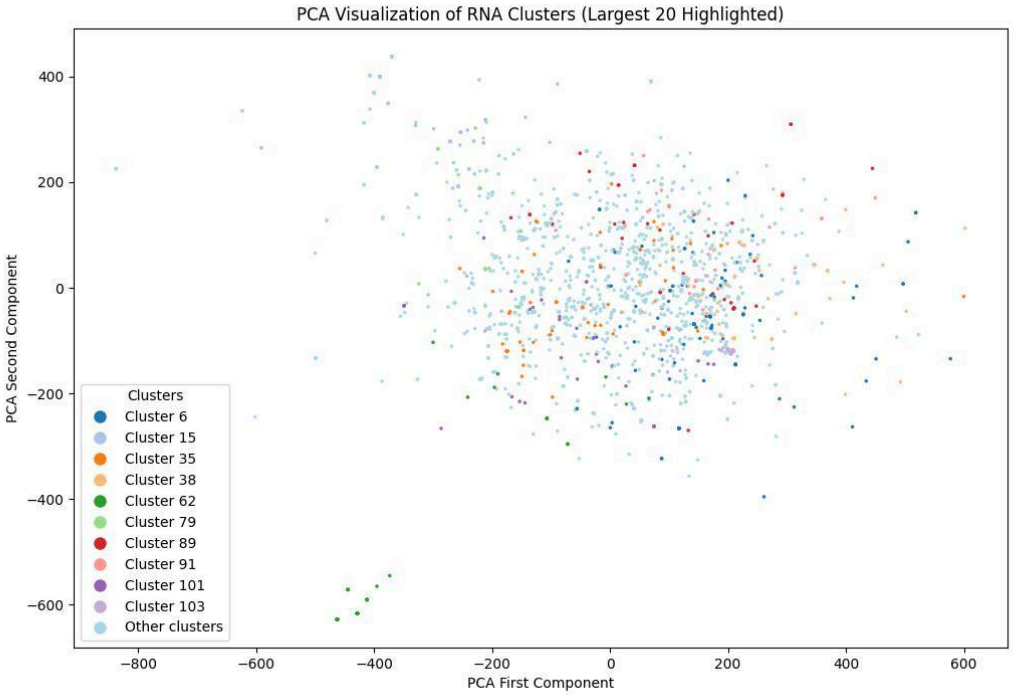


Together with the distance matrices of the RNA sequences, we use nucleotide letters ("A", "G", "T", "C") as one-hot encoder vectors with the following rules: A -> (1, 0, 0, 0), G -> (0, 1, 0, 0), T -> (0, 0, 1, 0), C -> (0, 0, 0, 1). We added padding with values (0, 0, 0, 0) to the sequences of length less than 64.

To avoid overfitting for the sequences with similar homology, we performed the homology filtering and clustering based on homology, which was similar to the approach in [30]. However, we used the

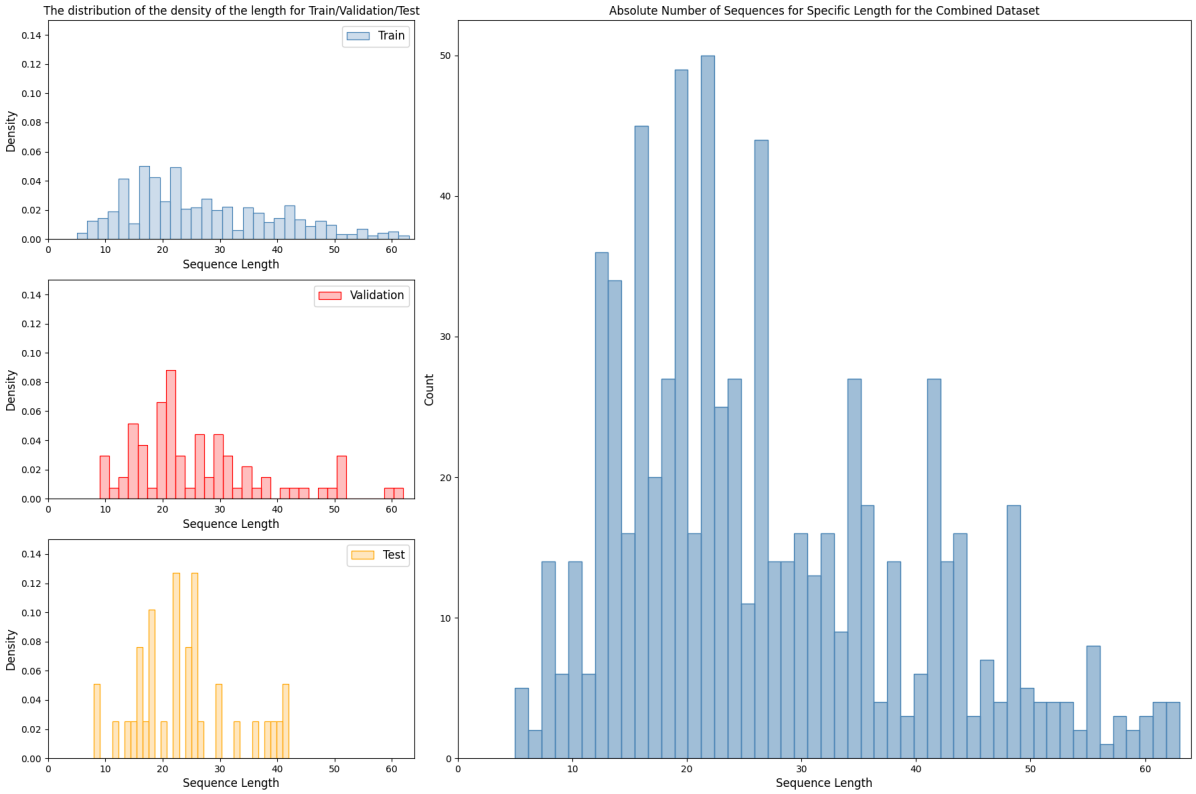
homology between the sequences as a differentiator between different clusters. Indeed, according to [31], homology is one of the options to cluster sequences into structural and functional clusters. We calculated the homology matrix for each pair of sequences. Then, we obtained the similarity matrix as a homology matrix subtracted from the identity matrix. If the element of the similarity matrix is close to zero, the sequences corresponding to that element are homologically similar. For clustering the RNA sequences, we used the Hierarchical Clustering algorithm since it allowed fitting to different shapes [33], which is crucial for the dataset with various homologies. The threshold of 60 has been chosen for the clustering cutoff level similar to [30]. This was in accordance with [32] within the interval of [60-70]. Additionally, we analysed the number of clusters as the result of the clustering. For the chosen number (threshold is 60) the number of clusters was 100. Whereas for the parameter of threshold of 70 the number of clusters was only 16, which was not enough for further splitting the data. The largest three clusters contain 111, 77, and 74 sequences. On Figure 2, the clustering of the RNA sequences is presented. We used the PCA algorithm to calculate the first two components and visualize the plot. Different colours correspond to different clusters found with the hierarchical clustering algorithm.

Figure 2. Two PCA components of the similarity matrix. Distinct colours represent distinct clusters obtained using Hierarchical Clustering.



The dataset was divided into [train | validation | test] subsets with an approximate proportion [0.80 | 0.15 | 0.05]. Particularly, each cluster from the hierarchical clustering obtained in the previous set was fully distributed either to the train, validation, or test subset with the probabilities of 0.80, 0.15, and 0.05 respectively. The total number of sequences was 842. For the [train | validation | test] after the partition, the numbers of sequences were [719 | 96 | 27]. In Figure 3 the distribution of the lengths of the sequences is presented for the [train | validation | test] subsets as well as for the combined dataset. We can see that the distributions are similar from the length's perspective, at least.

Figure 3. The probability density function of sequence lengths for the [train | validation | test] subsets (left) and the distribution of the sequences of the length for combined dataset (right).



Model

We used the variational autoencoder [17] to predict the distance matrix. Particularly, we have the conditions vector concatenated with the random vector as input. The condition vector is the flattened one-hot encoded nucleotide sequence (size of 256 nodes). The random greyscale image is a noise vector of the same size as the distance matrix (64*64 = 4096 nodes). We used a traditional variational autoencoder that consists of an encoder, bottleneck, and decoder. The encoder consists of two fully connected layers with a number of nodes equal to 4096 and 2048. The Bottleneck layer is trivial and consists of only one layer of size 1024. The decoder consists of three fully connected layers of size 2048, 4096, and 4096. The last layer represents the values of the distance matrix. After each fully connected layer (besides the last one), we used batch normalization, which improved the performance and added stability. In addition, we applied dropout with a rate of 0.5 for the output signals from the batch-normalization layers. The total number of trainable parameters in the model equals around 65M [64 532 480].

The loss function for the model is chosen to be classical for autoencoders and consists of two parts: the reconstruction loss and the Kullback–Leibler divergence loss [19]. We used Mean Average Error as a reconstruction loss due to its stability to outliers (in comparison with Mean Square Error Loss) [20]. We used the scaling parameter β [21] between two components as one of the hyper-parameters for the model. The loss function is presented below:

$$L_{\beta-VAE} = L_{RECON} + \beta \cdot L_{KL}$$

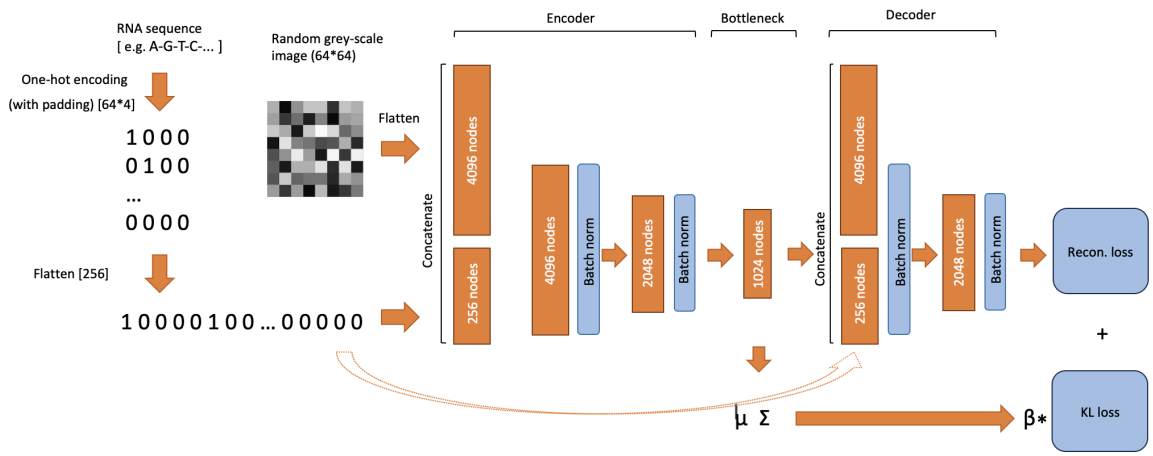
$$L_{MAE} = \sum\left(dm_{pred}[i] - dm_{true}[i]\right)$$

$$L_{KL} = -\frac{1}{2}\sum\left(1 + log\left(\sigma_j^2\right) - \mu_j^2 - e^{\sigma_j}\right)$$

Where L_{KL}, L_{MAE} are the Kullback-Leibler divergence and the reconstruction losses. σ_j, μ_j are the standard deviation and mean of the j -th dimension of the latent space.

We used mean average error (MAE) as a metric for the algorithm's performance. It is important to note that MAE was calculated only for significant values (the padding values were skipped while calculating the MAE and Loss function). The approximate scheme of the experiment is presented in Figure 4. The architecture was chosen after careful fine-tuning of the model.

Figure 4. The schematic representation of the VAE. The RNA sequences are on-hot encoded into numeric representation and fed together with the random image into the autoencoder. The variational autoencoder consists of 5 hidden layers: the encoder consists of two hidden layers (4096 and 2048 nodes); the bottleneck consists of a layer of 1024 nodes; the decoder consists of two layers (4096+256 and 2048 nodes). The loss function that is used is a combination of the reconstruction loss based on the RMSE and the KL loss.



It is worth mentioning that the distance matrices are supposed to be symmetric. We perform the symmetrization of the predicted distance matrices as a part of postprocessing as follows:

$$dm^{sym} = \frac{dm + dm^t}{2}$$

Indeed, we noticed that if the symmetrization is performed before the calculation of the loss function, the training becomes unstable. If we perform symmetrization before calculating the loss function, a set of values for the predicted distance matrix could satisfy the same value for the symmetric distance matrix. In other words, the sum would matter for the specific element of the symmetric matrix, making the values dm_{ij}, dm_{ji} more flexible by adding an extra degree of freedom. As a result, instability in training occurs. Therefore, the symmetrization was performed after the training loop as

a postprocessing technique for training subset. Also, for validation and test subsets the symmetrization was performed after the inference as the postprocessing technique.

Results

For the experiment, we trained our model for 2000 epochs. We used the validation set for fine-tuning the parameters, the test set was put aside only as a final prediction. We stored the model with the best validation MAE. For the training, the batch size was chosen to be 16. This parameter was chosen to make a solution stable, and the learning occurs fast enough. However, significant correlation of the batch size and final performance was not seen. For optimization, we used the Adam optimizer [22]. Regarding the limits of the number of nucleotides, the minimum and maximum lengths of the sequences were chosen for 4 and 64. We used the loss parameter balancing the KL loss and reconstruction loss as the one that gives the best performance and equals 1.1. The training time for 1000 epochs was around 2.1 hours (CPU Mac M1 processor). The inference time for one sequence during batch execution occurs to be around 0.001 seconds.

In addition to RMSE and MAE, we calculated the GDT-score and TM-score after symmetrization. The GDT score is calculated based on the distances between corresponding atoms (in our case – centres of masses) in the predicted and reference structures, evaluated at multiple distance thresholds. We used the values of 1, 2, 4, and 8 Å according to [34]. The TM-score was calculated based on [35] using tm-scoring python library [36]. The values of GDT and TM scores were included in the final computation due to the limitation of the RMSE (and MAE) metric for different lengths of the nucleotides. In Table 1, the results of the experiment are presented for the [train | validation | test] subset. The results (RMSE, MAE, RMSE+sym, MAE+sym) for the test dataset are slightly better than for the validation dataset due to the stochasticity of the values. Interestingly, the GDT-score is significantly lower for the test dataset than for the train and validation datasets. For the test set, we reached RMSE of 4.49Å, MAE of 3.13 Å. The overfitting is noticeable (due to the difference in the performance on the validation and train subset). Therefore, we suggest that having more data would only benefit the performance of this algorithm. Please note that we also included the performance results after postprocessing symmetrization. In particular, we used symmetrization on the predicted distance matrix. We can see that postprocessing only improves the algorithm's performance. Indeed, the RMSE and MAE with symmetrization are higher than without symmetrization. For the test dataset, the RMSE before symmetrization was 4.49 Å and after symmetrization 4.48 Å. However, the effect of symmetrization is minimal. It is worth mentioning that the test set was small (37 sequences), and the result that the performance on the test set is slightly better than on validation is fine due to high variance on the test set.

Table 1. The results of the VAE experiment for the Train | Validation | Test datasets. The metrics shown here are MAE, RMSE, MAE after symmetrization, RMSE after symmetrisation, TM-score after symmetrisation, GDT-score after symmetrisation. All the values are presented with the corresponding standard deviation values.

	MAE	**MAE+sym**	**RMSE**	**RMSE+sym**	**GDT-score**	**TM-score**
Train	3.23 ±1.64	3.22 ±1.64	4.69 ±2.46	4.69 ±2.46	39.67±20.58	0.25±0.14
Validation	3.86 ±2.61	3.86 ±2.61	5.56 ±3.77	5.56 ±3.77	26.83±20.87	0.18±00,5
Test	3.13±1.69	3.12±1.69	4.49±2.42	4.48±2.43	32.13±20.19	0.19±0.07

Figure 5 displays the computation results for two RNA sequences. Within each trio of images, the first two represent the actual and predicted distance matrices, respectively. The third image depicts the three-dimensional structure of the RNA molecule. We employed the distance geometry method to reconstruct the coordinates from the distance matrix, a classic technique proven effective in NMR experimental practices [23]. The challenge here is to find a set of coordinates that optimally fit the given distance constraints. According to [24], this problem can be approached by reformulating it as the task of finding a matrix

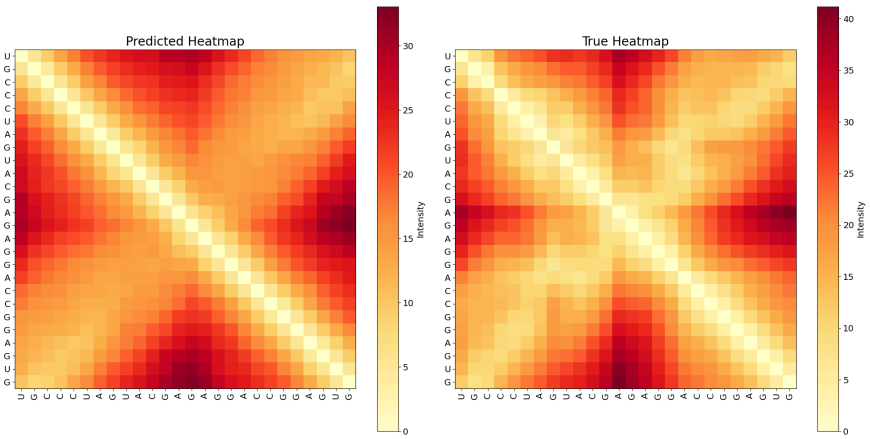
$$min\left|\left|C - G^2\right|\right|, \text{ where } G = -\frac{1}{2} H^T D H$$

This problem could be solved via the computation of the spectral decomposition of distance matrices and computation of eigenvalues and eigenvectors. The components of this matrix are the coordinates we need.

In addition to the basic experiments, we decided to check the variational idea of the autoencoder. We generated the distance matrices five times for several sequences using different noise vectors. In Figure 6, the results of the experiment are presented. We can see that the change in noise vectors does not significantly alter the predicted distance matrix. In other words, the variational factor of the algorithm has minimal influence on the outcome. Let us calculate the average coefficient of variation as a ratio between the standard deviation of the error for the different noise vectors for all the test sequences. Indeed, the ratio is small.

$$cv = \frac{\sigma}{\mu} = 0.13$$

Figure 5. Distance matrices and 3d structure for three RNA sequences (true and predicted). The left image represents the Predicted heatmap, the right image represents the True heatmap.



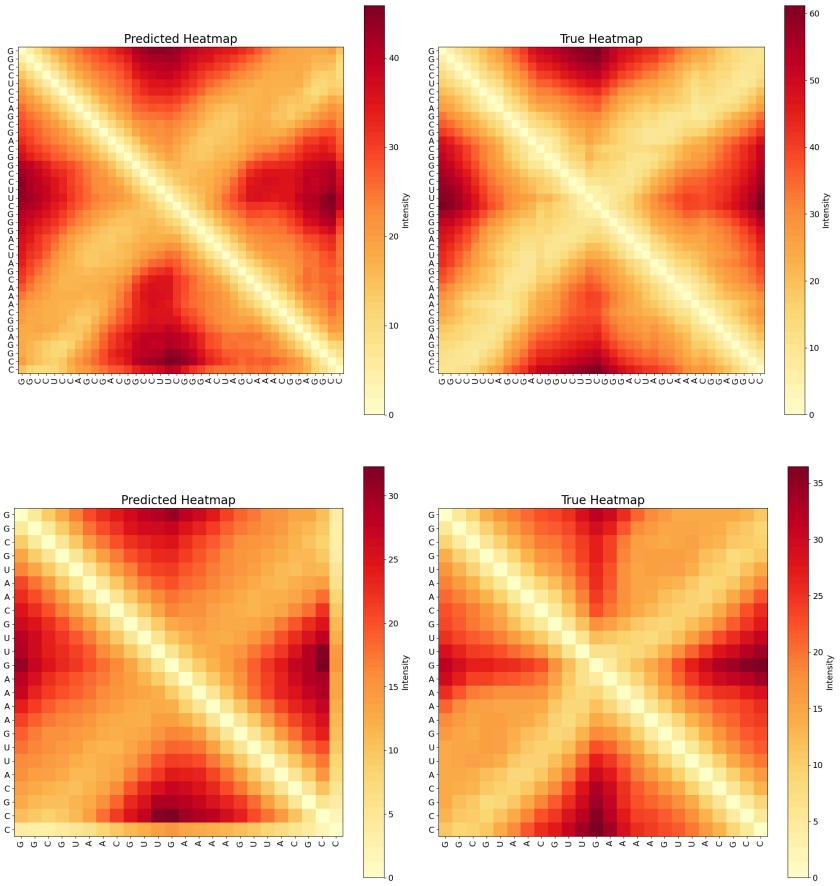
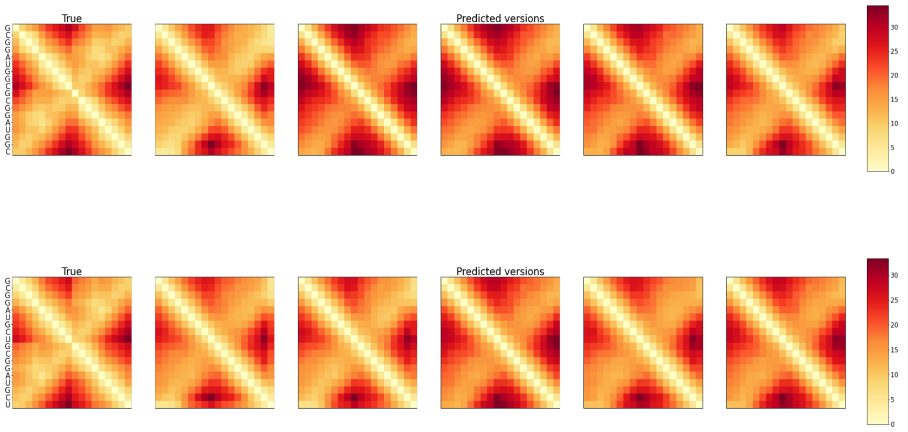
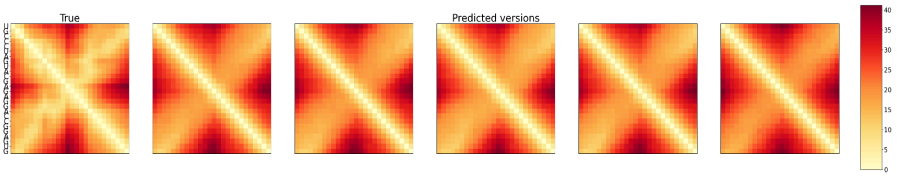


Figure 6. Generated structures with different noise vectors for three RNA structures. The left images represent the True values, the Five Right images are the predicted structures for different initial noise vectors. The nucleotide labels are shown on the left.





In the next experiment, we trained the same model but for larger nucleotides. In particular, we fixed the maximum length of the nucleotides to 128 while other parameters were not changed. The [train | validation | test] subsets were different from the subsets where the maximum length was fixed to 64. Therefore, we should compare the results with Table 1 considering the stochastic component. We communicate the results of the experiments on the test datasets in Table 2.

Table 2. The comparison of the performance of AutoRNA algorithm for small and large RNA sequences for the test and train datasets

	Test		
	Number of sequences	RMSE	GDT-score
Sequences, less than 32 b.p	30	4.49± 2.35	32.13 ± 20.19
Sequences, less than 64 b.p and more than 32 b.p.	7	6.05± 1.17	14.72 ± 4.33

For estimation of the quality of our 3D structure prediction, we compared our algorithm with RhoFold [25] and SimRNA [16]. For the simulation in SimRNA and RhoFold, we used a standard parameter preset with 10000 cycles. The comparison was provided for the random subset, which was not used in the training (or validation). For RMSD computation between experimental and modelled structures, we used the PyMol [26] package. The results are presented in Table 3.

Table 3. The comparison of existing algorithms with the developed algorithms.

	Our algorithm	**Rhofold**	**SimRNA**
MAE (Å), less than 32 b.p	3.475	**3.414**	3.990
MAE (Å), over 32 b.p.	4.248	**3.692**	5.125

Postprocessing

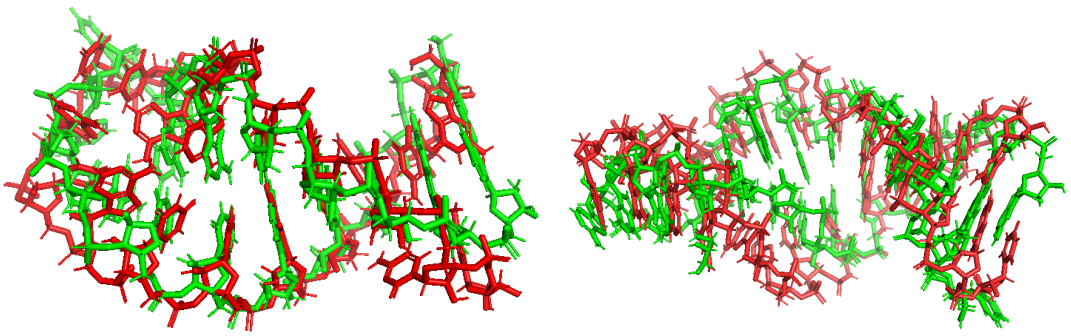
The full inference pipeline consists of the following steps:

1. VAE center-of-mass prediction
2. Structure assembly from templates
3. Correcting obtaining structure, adding lacking atoms
4. Molecular dynamics for polishing structure and removing of clashes between atoms

After receiving predictions of the center of mass, we must restore the rest of the RNA structure. For that we use a base of templates from PDB. We extract from PDB structures for all possible fourths of nucleotides. After obtaining the distance matrix from our VAE, we use distance geometries to convert it into 3D coordinates of RNA molecules. We used the library's most suitable templates and assembly structure. That library was assembled from the whole PDB RNA. For each RNA molecule in the PDB, all possible quadruplets of nucleotides were selected and put into their respective catalogs. Accordingly, 256 catalogs were created, in which 46199 files were collected. For each nucleotide in each template, we have computed the centres of mass and align obtained centres of mass of template and prediction into our results. This approach helps us to reduce the number of parameters of VAE.

In the final stage, we correct the covalent bound and flat angles that arise in the resulting RNA structure. For this we run molecular dynamics computations and make optimization of potential energy receiving RNA molecules. As a molecular dynamics engine we use the OpenMM package [37]. A similar solution for structure refinement was implemented as the final stage of AlphaFold2 pipeline [2]. Molecular dynamics was run on 1M steps by solving Langevin dynamic equations. As an atom forcefield we use the AMBER14 forcefield. All computation was performed in standard conditions (300 K, 7 pH, 1 atm.). Modelling was performed in a water cube with a 3-tip water model with 2 nm padding from RNA molecules. Protonation of RNA molecules was performed by pdb4amber software. Also, that software fills the lacking atoms in some templates.

Figure 7. Comparison between structure of RNA stem loop 6XXB and 7L3R PDB (left and right) structures for experimental and predicted structure. Green color corresponds to the actual structure, red color corresponds to the predicted structure.



Discussion

The proposed approach has several shortcomings despite its simplicity and robustness. The main drawback is that the predicted and learning set distance matrices have a fixed size. In addition, the size cannot be very large. Indeed, the PDB dataset contains not enough long RNA structures which is not sufficient for proper training.

The presented architecture of the autoencoder is one of the possible approaches for RNA folding problem-solving. Another possible approach for improving quality prediction could be to use graph neural networks. Also, a very promising path would be adding several RNA-specific features into the

model based on the biophysical properties of molecules. For example, as an additional feature, we can consider the secondary structure of RNA [28], the functional class of predicted molecules or features, obtained from RNA templates. As shown in protein cases [2], significant improvement in model quality could be achieved by the addition of evolutionary information [28]. But, instead of a protein folding problem, RNA molecules do not have a good model of nucleotide coevolution and its connection to RNA tertiary structure. Therefore, before adding this coevolution information, we must understand the applicability of such a model to our problem. In addition, we can use physically informed Neural Networks [18] to gain additional information about the interaction of atoms. An alternative way to improve the performance is to find a feasible way to augment data. At the moment, the data for the large molecules is limited. Modeling new RNA structures by physical-based methods (Molecular Dynamics, Monte Carlo Sampling) could give us good-quality structures but will be slow and computationally expensive. A practical solution could be to use the rotamers library. We can change one or several nucleotides in RNA into other nucleotides. Their local conformation must be in accordance with the rotamer library. After that, we can perform several Molecular Dynamics steps to remove the clash and improve the structure.

Even though our model is a “black box”, the predictor consistently returns physically plausible structures. Let us take an example of a 1KFO molecule. In the database, this RNA is listed as a single strand with no annotated secondary or tertiary structure. When we supply only the primary sequence to our model—without any secondary-structure constraints—it spontaneously folds the molecule into a canonical hairpin. This conformation agrees with established principles of RNA folding. Therefore, the model can highlight potential mis-annotations in existing structural databases.

Conclusion

In this study, we introduce a novel approach for predicting RNA tertiary structure based on a Variational Autoencoder (VAE) architecture. Unlike traditional computational methods such as Molecular Dynamics simulations [29], our method leverages the complex, non-linear relationships embedded in RNA sequences to generate accurate structural predictions. This data-driven approach enables efficient modeling of RNA folding mechanisms and offers a scalable alternative to physics-based simulations, which are often computationally intensive.

Understanding the three-dimensional conformation of RNA is critical for numerous biomedical applications, including the design of RNA-based therapeutics and the development of next-generation RNA vaccines. Our method demonstrates a promising step in this direction: we achieved a root-mean-square error (RMSE) of approximately 4.5 Ångströms when predicting the center-of-mass positions of individual nucleotides, highlighting the potential of deep generative models in structural bioinformatics

The algorithm still has some limitations regarding the specific RNA sequences due to the need for appropriate training data. For some structures, sub-optimal conformations that could vary from the original tertiary structures are found. The presented algorithm could be used to predict the RNA tertiary structure (the algorithm can be found via the link https://github.com/quantori/AutoRNA)

Contributions

M.K. contributed to planning, discussion, pre-processing the dataset, creating models, creating visualisations, performing experiments, writing manuscript.

L.U contributed to planning, discussion, pre-processing the dataset, reconstruction of a 3d structure, and writing the manuscript.

F. Z. contributed to pre-processing the dataset, discussion, creating visualization

I. P. contributed to discussion, creating visualizations

O. K. contributed to the discussion, writing a manuscript.

Y. G. contributed to planning, discussion, and organizing the research.

Acknowledgements: A. Kasianov, A. Krasnaya, S. Semyonov, A. Safaraleev

Data availability

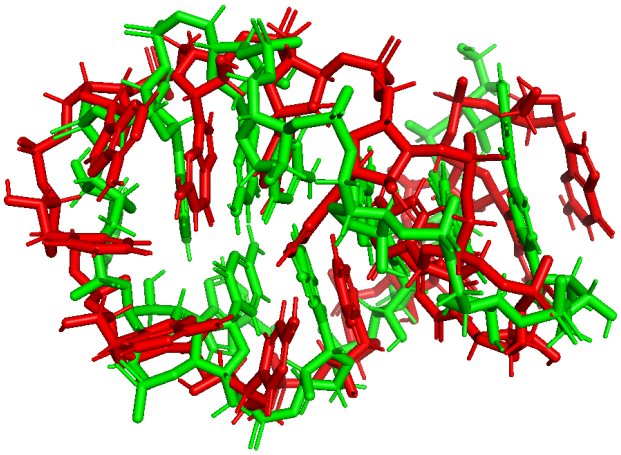
The data and code that support the findings of this study are available in Zenodo and Github correspondingly. The code used for this research can be accessed via the link https://github.com/quantori/AutoRNA. The dataset used in this research can be accessed under the https://doi.org/10.5281/zenodo.16887142.

Supplementary material

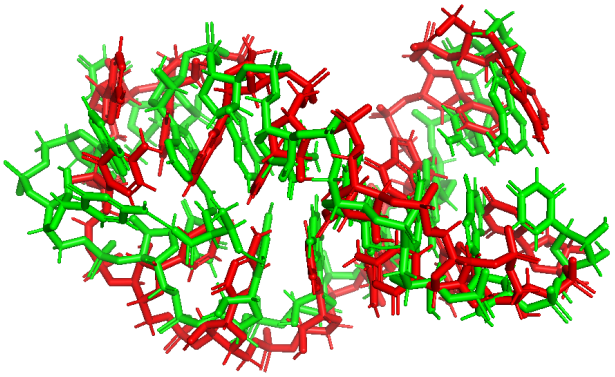
Nucleotide predictions (additional examples).

** Green color reflects the actual structure, red color reflects the predicted structure.*

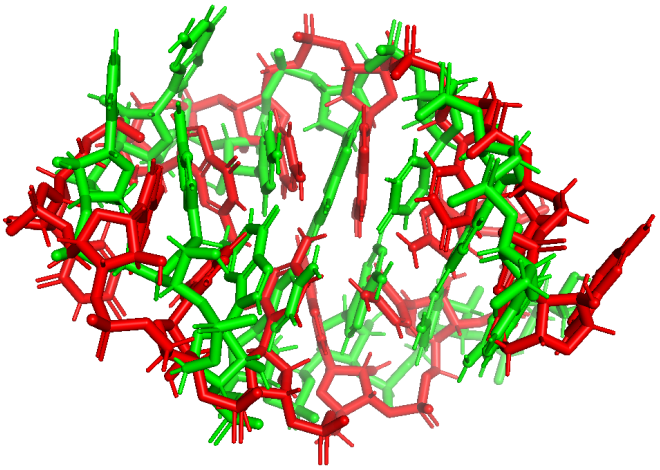
1ESH



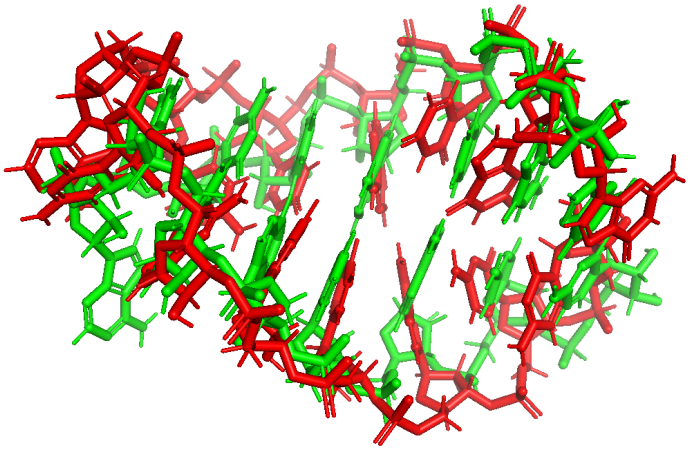
2G1G



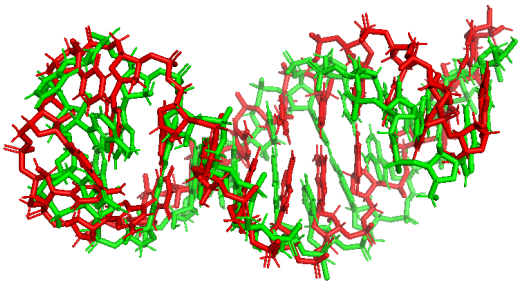
1I46



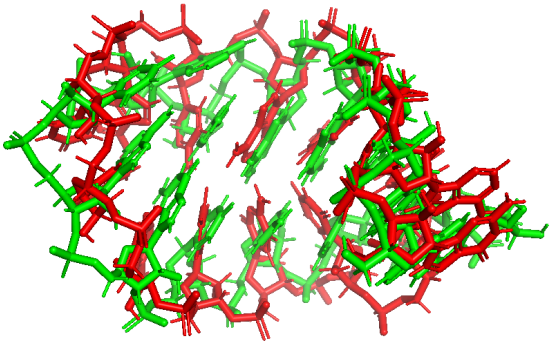
1ROQ



1NZ1



2LPA



References

[1] Yu A.M., Choi Y.H., Tu M.J. (2020) RNA drugs and RNA targets for small molecules: principles, progress, and challenges. *Pharmacological Rev.*, 72(4), 862-898.

[2] Jumper J., Evans R., Pritzel A., Green T., Figurnov M., Ronneberger O., Tunyasuvunakool K., Bates R., Žídek A., Potapenko A., Bridgland A., Meyer C., Kohl S.A.A., Ballard A.J., Cowie A., Romera-Paredes B., Nikolov S., Jain R., Adler J., Back T., Petersen S., Reiman D., Clancy E., Zielinski M., Steinegger M., Pacholska M., Berghammer T., Bodenstein S., Silver D., Vinyals O., Senior A.W., Kavukcuoglu K., Kohli P., Hassabis D. (2021) Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873), 583-589.

[3] Burley S.K., Berman H.M., Kleywegt G.J., Markley J.L., Nakamura H., Velankar S. (2017) Protein Data Bank (PDB): The single global macromolecular structure archive. In: Read R.J., Schmidt H. (eds). *Methods in Molecular Biology*. Springer, New York, pp. 627-641. https://doi.org/10.1007/978-1-4939-7000-1_26.

[4] Gillespie J., Mayne M., Jiang M. (2009) RNA folding on the 3D triangular lattice. *BMC Bioinformatics*, 10, 1-17.

[5] Xia Z., Gardner D.P., Gutell R.R., Ren P. (2013) RNA 3D structure prediction by using a coarse-grained model and experimental data. *J. Phys. Chem. B.*, 117(11), 3135-3144.

[6] Tan Y.L., Liu H., Tan Z.J., Shi Y., Zhou P., Lu Z.J. (2022) rsRNASP: A residue-separation-based statistical potential for RNA 3D structure evaluation. *Biophys. J.*, 121(1), 142-156.

[7] Rahmanov S., Kulakovskiy I., Uroshlev L., Makeev V. (2010) Empirical potentials for ion binding in proteins. *Journal of Bioinformatics and Computational Biology*, 8(03), 427-435.

[8] Case D.A., Cheatham III T.E., Darden T., Gohlke H., Luo R., Merz Jr K.M., Woods R.J. (2005) The Amber biomolecular simulation programs. *Journal of Computational Chemistry*, 26(16), 1668-1688.

[9] Sharma S., Ding F., Dokholyan N.V. (2008) iFoldRNA: three-dimensional RNA structure prediction and folding. *Bioinformatics*, 24(17), 1951-1952.

[10] Van Der Spoel D., Lindahl E., Hess B., Groenhof G., Mark A.E., Berendsen H.J. (2005) GROMACS: fast, flexible, and free. *Journal of Computational Chemistry*, 26(16), 1701-1718.

[11] Al-Hashimi H.M., Walter N.G. (2008) RNA dynamics: it is about time. *Current Opinion in Structural Biology*, 18(3), 321-329.

[12] Wang W., Feng C., Han R., Wang Z., Ye L., Du Z., Yang J. (2023) trRosettaRNA: automated prediction of RNA 3D structure with transformer network. *Nature Communications*, 14(1), 7266.

[13] Li J., Chen S.J. (2021) RNA 3D structure prediction using coarse-grained models. *Frontiers in Molecular Biosciences*, 8, 720937.

[14] Watkins A.M., Rangan R., Das R. (2020) FARFAR2: improved de novo rosetta prediction of complex global RNA folds. *Structure*, 28(8), 963-976.

[15] Fu L., Cao Y., Wu J., Peng Q., Nie Q., Xie X. (2022) UFold: fast and accurate RNA secondary structure prediction with deep learning. *Nucleic Acids Research*, 50(3), e14-e14.

[16] Bugnon L.A., Di Persia L., Gerard M., Raad J., Prochetto S., Fenoy E., Milone D.H. (2023) sincFold: end-to-end learning of short-and long-range interactions for RNA folding. *bioRxiv*, 2023-10.

[17] Doersch C. (2016) Tutorial on Variational autoencoders. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1606.05908.

[18] Cuomo S., Di Cola V.S., Giampaolo F., Rozza G., Raissi M., Piccialli F. (2022) Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3), 88.

[19] Asperti A., Trentin M. (2020) Balancing reconstruction error and kullback-leibler divergence in variational autoencoders. *IEEE Access*, 8, 199440-199448.

[20] Dixon W.J. (1953) Processing data for outliers. *Biometrics*, 9(1), 74-89.

[21] Burgess C.P., Higgins I., Pal A., Matthey L., Watters N., Desjardins G., Lerchner A. (2018) Understanding disentangling in beta-VAE. *arXiv.org*. https://arxiv.org/abs/1804.03599.

[22] Kingma D.P., Ba J. (2014) Adam: A method for stochastic optimization. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1412.6980.

[23] Mucherino A., Papajorgji P.J., Pardalos P.M. (2012) Distance geometry: theory, methods, and applications. Springer Science & Business Media.

[24] Trosset M.W. (1998) A new formulation of the nonmetric strain problem in multidimensional scaling. *J. Classif.*, 15(1), 15-35.

[25] Shen T., Chen J., Li Y., Tang L., Huang Y., He J., Wu Z., Xu J., Cheng J. (2022) E2Efold-3D: end-to-end deep learning method for accurate de novo RNA 3D structure prediction. *arXiv preprint arXiv:2207.01586*.

[26] DeLano W.L., Ultsch M.H., Palo Alto C.A. (2002) Pymol: An open-source molecular graphics tool. *CCP4 Newsl. Protein Crystallogr.*, 40(1), 82-92.

[27] Ho J., Jain A., Abbeel P. (2020) Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33, 6840-6851.

[28] Vicens Q., Kieft J.S. (2022) Thoughts on how to think (and talk) about RNA structure. *Proc. Natl. Acad. Sci. U.S.A.*, 119(17), e2112677119.

[29] Šponer J., Bussi G., Krepl M., Banáš P., Bottaro S., Cunha R.A., Gil-Ley A., Pinamonti G., Poblete S., Jurečka P., Walter N.G., Otyepka M. (2018) RNA structural dynamics as captured by

molecular simulations: A comprehensive overview. *Chem. Rev.*, 118(8), 4177-4338. https://doi.org/10.1021/acs.chemrev.7b00427.

[30] Hong, X., Zheng, J., Xie, J., Tong, X., Liu, X., Song, Q., Liu, S., & Liu, S. (2021). RR3DD: an RNA global structure-based RNA three-dimensional structural classification database. *RNA Biology*, *18*(sup2), 738–746. https://doi.org/10.1080/15476286.2021.1989200

[31] Rivas, E. (2020). RNA structure prediction using positive and negative evolutionary information. *PLoS Computational Biology*, *16*(10), e1008387. https://doi.org/10.1371/journal.pcbi.1008387

[32] Li, W., & Godzik, A. (2006). Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13), 1658–1659. https://doi.org/10.1093/bioinformatics/btl158

[33] Petegrosso, R., Li, Z. and Kuang, R. (2019). Machine learning and statistical methods for clustering single-cell RNA-sequencing data. *Briefings in Bioinformatics*. doi:https://doi.org/10.1093/bib/bbz063.

[34] Zemla, A. (2003). LGA: a method for finding 3D similarities in protein structures. *Nucleic Acids Research*, *31*(13), 3370–3374. https://doi.org/10.1093/nar/gkg571

[35] Zhang, Y. (2005). TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic Acids Research*, *33*(7), 2302–2309. https://doi.org/10.1093/nar/gki524

[36] *tmscoring*. (2019, March 22). PyPI. https://pypi.org/project/tmscoring/

[37] Eastman, P., Swails, J., Chodera, J.D., McGibbon, R.T., Zhao, Y., Beauchamp, K.A., Wang, L.P., Simmonett, A.C., Harrigan, M.P., Stern, C.D., Wiewiora, R.P., Brooks, B.R., & Pande, V.S. (2017). *OpenMM 7: Rapid development of high performance algorithms for molecular dynamics*. PLOS Computational Biology, **13**(7), e1005659. https://doi.org/10.1371/journal.pcbi.1005659