Uncovering the Folding Landscape of RNA Secondary Structure with Deep Graph Embeddings

Egbert Castro [1] **Andrew Benz** [2] **Alexander Tong** [3] **Guy Wolf** [*4] **Smita Krishnaswamy** [*3 5]

Abstract

Biomolecular graph analysis has recently gained much attention in the emerging field of geometric deep learning. Here we focus on organizing biomolecular graphs in ways that expose meaningful relations and variations between them. We propose a geometric scattering autoencoder (GSAE) network for learning such graph embeddings. Our embedding network first extracts rich graph features using the recently proposed geometric scattering transform. Then, it leverages a semi-supervised variational autoencoder to extract a low-dimensional embedding that retains the information in these features that enable prediction of molecular properties as well as characterize graphs. We show that GSAE organizes RNA graphs both by structure and energy, accurately reflecting bistable RNA structures. Also, the model is generative and can sample new folding trajectories.

1. Introduction

While RNA is sometimes thought of as a linear sequence of bases, non-coding RNA especially can fold into 3D structure that has functionality (Ganser et al., 2019). Each RNA sequence has the propensity of folding into many different structures transiently, but fewer structures stably. In general, it helps explore the functionality of RNA structures if we can embed them in ways that uncover features of their folding landscape, and smoothly reflect their transitions. This motivates the examination of graph embeddings generated by neural networks to see if they can organize RNA graphs into coherent landscapes or folding manifolds. Using such embeddings, biologists could, for example, assess the likelihood that the RNA could switch from one structure to another to change functionality, i.e., whether it is a riboswitch – a question that is difficult to answer normally (Schroeder, 2018).

Our embedding procedure based on three goals: • Obtaining faithful embeddings where neighbors are close in both in terms of graph structure and in terms of molecular properties. • Enabling visual exploration and interpretation of biomolecular structures using the embedding space. • Generating trajectories of folds and decode them into molecular folds. We define these desiderata more formally in Section 3.

Here, we propose a new framework for organizing biomolecular structures called geometric scattering autoencoder (GSAE). First, GSAE encodes molecular graphs based on scattering coefficients (Gao et al., 2019; Gama et al., 2019a) of dirac signals placed on their nodes. Next, it uses an autoencoder architecture to further refine and organize the scattering coefficients into a reduced and meaningful embedding based on both a reconstruction penalty and auxiliary penalties to predict molecular properties. Finally, to generate graphs we train a scattering inversion network (SIN) that takes scattering coefficients as inputs and generates adjacency matrices.

1.1. Contributions

In this work (1) we introduce graph scattering transforms to an autoencoder framework in the form of GSAE, (2) we demonstrate that GSAE can be used for the faithful embedding and visualization spaces of RNA structures as well as synthetic graphs, (3) we show that SIN allows for the generation of quasi-trajectories in the RNA folding domain. We compare our results to several of the most prominent GNN-based graph representation approaches including GAE (Kipf and Welling, 2016a), GVAE (Kipf and Welling, 2016a), as well as non-trainable methods like embeddings of the WL-kernel computed on graphs (Shervashidze et al., 2011) or embeddings of graph edit distance matrices on toy and RNA datasets.

[*]Co-supervised [1]Comp. Bio. and Bioinf. Program, Yale University; [2]Dept. of Mathematics, Yale University; [3]Dept. of Computer Science, Yale University; [4]Dept. of Mathematics and Statistics, Université de Montréal; Mila – the Québec AI institute; [5]Dept. of Genetics, Yale University. Correspondence to: Smita Krishnaswamy <smita.krishnaswamy@yale.edu>.

Presented at the *ICML 2020 Graph Representation Learning and Beyond (GRL+) workshop*.

2. Background

The geometric scattering transform (Gao et al., 2019; Gama et al., 2019b) is based on a cascade of graph wavelets, typically constructed via diffusion wavelets (Coifman and Maggioni, 2006). These are constructed using a lazy random walk diffusion operator $\mathbf{P} = \frac{1}{2}(\mathbf{I} + \mathbf{A}\mathbf{D}^{-1})$, where A is the adjacency matrix of the analyzed graph and D is a diagonal matrix of its vertex degrees. Then, P^t, $t > 0$, contains t-step diffusion transition probabilities between graph nodes. These powers of P can also be interpreted as lowpass filters that average signals over multiscale diffusion neighborhoods, where the size (or scale) of the neighborhood is determined by t. Therefore, given a graph signal f, the filtered signal $\mathbf{P}^t f$ only retains intrinsic low frequencies over the graph. Similarly, $I - P^t$, $t > 0$, form a highpass filters whose scales is determined by t. The diffusion wavelets transform (Coifman and Maggioni, 2006) combines these lowpass and highpass filters to form bandpass filters of the form $\mathbf{\Psi}_j = \mathbf{P}^{2^{j-1}} - \mathbf{P}^{2^j} = \mathbf{P}^{2^{j-1}}(\boldsymbol{I} - \mathbf{P}^{2^{j-1}})$, with dyadic scales 2^j, $j = 1, \ldots, J$ where J defines the widest scales considered (corresponding to 2^J random walk steps). The resulting wavelet transform then yields wavelet coefficients $\boldsymbol{\mathcal{W}} f = \{\mathbf{P}^t f,\ \mathbf{\Psi}_j f\}_{j=1}^{\log_2 t}$ that decompose f into a family of signals that capture complementary aspects of f at different scales (i.e., intrinsic frequency bands on the graph).

While the wavelet coefficients $\boldsymbol{\mathcal{W}} f$ give a complete and invertible representation of f, the representation provided by $\mathbf{\Psi}_j f$ is not guaranteed to provide stability or invariance to local deformations of the graph structure. To obtain such representation, Gao et al. (2019) propose to follow the same approach as in expected scattering of traditional signals (Mallat, 2012; Bruna and Mallat, 2013) to aggregate wavelet coefficients by taking statistical moments after applying nonlinearity in form of absolute value. Their first-order scattering features are $S_1 f = [\| \, |\mathbf{\Psi}_j f| \, \|_q]_{1 \le j \le J, 1 \le q \le Q}$, which capture the statistics of signal variations over the graph. They are complimented on one hand by zeroth-order scattering, consisting of statistical moments of f itself (without filtering), and on the other hand with higher order scattering coefficients that capture richer variations eliminated by the aggregation in the above equation. In general, m^{th} order scattering features are computed by a cascade of m wavelet transforms and absolute-value nonlinearities, creating a designed (i.e., non-learned) multiscale graph neural network: $S_m[j_1, \ldots, j_m, q] f = \| \, |\mathbf{\Psi}_{j_m}| \cdots |\mathbf{\Psi}_{j_1} f| \cdots | \, \|$, with features indexed by moment q and scales $j_1, \ldots, j_m$. Due to the multiresolution nature of these features, they provide a rich and stable description of f (see Gao et al., 2019; Perlmutter et al., 2019; Gama et al., 2019a;b, for more details).

3. Problem setup

Given a set of graphs $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$, we aim to find an embedding $Z_{\mathcal{G}} = \{z_1, z_2, \ldots, z_n\}$ in Euclidean space, i.e., where each graph G_i is mapped to a d-dimensional vector $z_i \in \mathbb{R}^d$, where the embedding satisfies the following properties, which we will validate empirically for our proposed construction: 1. **Faithfulness:** the embedding should be faithful to the graphs in G in the sense that graphs that are near each other in terms of graph edit distance should be close to each other in the embedding space, and vice versa. Formally we aim for $\|z_i - z_j\| < \epsilon$, for some small ϵ, to be (empirically) equivalent to $ged(G_i, G_j) < \nu$ for some small ν where ged is graph edit distance. 2. **Smoothness:** the embedding should be smooth in terms of a real valued meta-property $M = \{m_1, m_2, \ldots, m_n\}$, where $m_i \in \mathbb{R}^n$, which is only given on the training data. 3. **Invertibility:** it should be possible to generate new graphs by interpolating points in the embedded space and then inverting them to obtain interpolated graphs between training ones. Formally, for any two points z_x, z_y in the embedding space, we expect $z = (z_x + z_y)/2$ to match the embedding of a valid graph, with properties specified in the previous criteria, and with a constructive way to (approximately) reconstruct this graph.

To explain the second criterion, given an affinity matrix of vectors in $Z_{\mathcal{G}}$, denoted $A_{Z_{\mathcal{G}}}$, where $A_{\mathcal{G}}(i,j) = similarity(z_i, z_j)$, we define a Laplacian matrix of this embedding as $L = D - A_{\mathcal{G}}$ where D is a diagonal matrix whose entry $D(i,i) = \sum_j A(i,j)$, we want the *dirichlet energy* $M^T L M$ to be small. However, the difficulty in biological graphs is that M is an emergent property that can be difficult to compute from the graph. In principle, this smoothness could be enforced for multiple meta properties.

4. Geometric Scattering Autoencoder

To derive an embedding that has the properties described in the previous section, we propose a novel framework based on the untrained geometric scattering, a trained autoencoder, and a scattering inversion network, as shown in Figure 1.

The first step in our construction is to extract scattering features from an input graph, thus allowing us to further process the data in a Euclidean feature space. Since the biomolecule graphs considered in this work do not naturally provide us with graph signals, we have to define characteristic signals that will reveal the intrinsic graph structure. However, since we mostly focus here on RNA folding applications, we assume there is node correspondence between graphs, and thus we can produce a set of diracs $d_i = \{0, \ldots, 1, \ldots, 0\}$ that provide one-hot encoding of each node v_i in the graph (i.e., $d_i[j] = 1$ iff $i = j$; zero otherwise).

Next, we map an input graph to a Euclidean feature space given by the scattering features of these dirac signals over

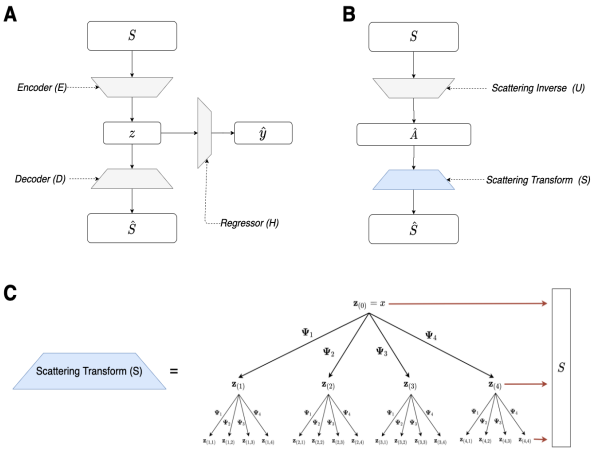

Figure 1. A. GSAE, B. Inverse transform transform network. C. Scattering Transform network (S)

the graph. For each dirac signal d_i, we take the zeroth, first and second order scattering and concatenate them across orders. Then, we concatenate the scattering coefficients of all the dirac signals over the graph to obtain its entire scattering feature vector. Formally, let Π denote the concatenation operator, then this feature vector is given by $S(G) = \Pi_{i=1}^{n}\Pi_{m=0}^{2}S_m^{(G)}d_i$ is constructed using graph wavelets from a lazy random walk over the graph G, where the superscript indicates the scattering operation.

The scattering representation provided by $S(G)$ encodes the graph geometry in a Euclidean feature space that is high dimensional and often highly redundant. Indeed, as shown in (Gao et al., 2019), it is often possible to reduce significantly the dimensionality of scattering representations while still maintaining the relations between graphs encoded by them. Therefore, the next step in our embedding construction is to apply an autoencoder to the scattering features in $S(G)$. Formally, we train an encoder $E(\cdot)$ and decoder $D(\cdot)$ such that $\hat{S}(G) = D(E(G))$ will approximately reconstruct $S(G)$ via a MSE penalty $\|S(G) - \hat{S}(G)\|^2$. However, as mentioned in Sec. 3, in addition to the unsupervised information captured and provided by $S(G)$, we also aim for our embedding to follow physical properties of the biomolecules represented by the graphs. These are encoded by meta properties available at the graph level, denoted here by $m(G)$. Therefore, in addition to the reconstruction penalty, we also introduce a supervised penalty in the loss for predicting $m(G)$ via an auxiliary network $H(\cdot)$ operating on the latent embedding. Formally, this penalty is added to the autoencoder loss via a term $\|m(G) - H(E(S(G)))\|^2$.

Finally, since we aim for our embedding to be approximately invertible, we must also construct a transform that maps embedded representations into viable graphs. We recall that our data consists of graphs that all share the same nodes, and therefore this construction is only required to infer an adjacency matrix from embedded coordinates. The autoencoder trained in the previous step naturally provides a decoder that (approximately) inverts the latent representation into geometric scattering features. Furthermore, to ensure stability of this inversion to perturbation of embedded coordinates, as well as enable (re)sampling from the embedding for generative purposes, we add VAE loss terms to our autoencoder, injecting noise to its latent layer and regularizing its data distribution to resemble normal distribution via KL divergence as in Kingma and Welling (2013).

Table 1. Results show structural organization of the various embeddings on the two bistable datasets. Graph dirichlet energy with respect to the graph edit distance from the two stable energy minima are reported. Here "+ H" refers to the addition of the energy prediction auxilliary network H

	SEQ3		SEQ4	
	Min 1	Min 2	Min1	Min 2
GED	0.442 ± 0.0003	0.517 ± 0.0002	0.045 ± 0.0003	0.058 ± 0.0003
Scat. Coeff.	0.0604 ± 0.0003	0.0732 ± 0.0002	0.066 ± 0.0002	0.0859 ± 0.0005
GAE	0.035 ± 0.001	0.045 ± 0.002	0.038 ± 0.001	0.053 ± 0.003
GAE + H	0.044 ± 0.006	0.06 ± 0.006	0.043 ± 0.003	0.062 ± 0.003
VGAE	0.425 ± 0.006	0.478 ± 0.008	0.443 ± 0.007	0.528 ± 0.008
VGAE + H	0.392 ± 0.005	0.46 ± 0.008	0.405 ± 0.006	0.469 ± 0.006
WL-Kernel	0.185 ± 0.0012	0.225 ± 0.001	0.2 ± 0.0016	0.263 ± 0.0016
GSAE - AE	0.069 ± 0.001	0.087 ± 0.002	0.069 ± 0.001	0.085 ± 0.002
GSAE (no H)	0.337 ± 0.021	0.381 ± 0.027	0.112 ± 0.004	0.038 ± 0.001
GSAE	0.346 ± 0.076	0.402 ± 0.074	0.103 ± 0.004	0.124 ± 0.005

Our final step is to construct a scattering inversion network (SIN) that is able to construct adjacency matrices from scattering features. We observe that the main challenge in optimizing such an inversion network is how to define a suitable loss on the reconstructed adjacency matrices. We mitigate this by leveraging the geometric scattering transform itself to compute the inversion loss. Namely, we treat the concatenated construction of $S(\cdot)$ as a decoder and then train the inversion network $U(\cdot)$ as an encoder applied to $S(G)$ such that the scattering features of the resulting graph will approximate the input ones, penalized via the MSE: $\|S(G) - S(U(S(G))\|^2$.

Putting all the components together, the geometric scattering autoencoder (GSAE) trains four networks (E,D,H,U) with a combined loss: $\mathbb{E}_{G\in\mathcal{G}}\|D(E(S(G))) - S(G)\|^2 + \alpha\|H(S(G)) - m(G)\|^2 + \beta\|S(G) - S(U(S(G))\|^2$, where α and β are tuning hyperparameters controlling the importance of each component in the loss.

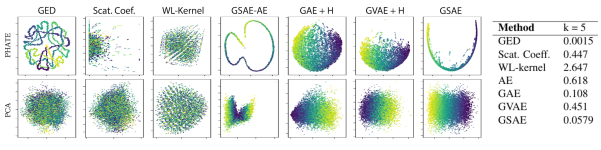

Figure 2. A. PHATE and PCA plots of seven different embeddings of the random graph dataset. Color corresponds to the position in the 10,000-step sequence of graphs, the ordering of which GSAE reveals clearly. B. Graph dirichlet energy with respect to the step indices of the trajectory sequence.

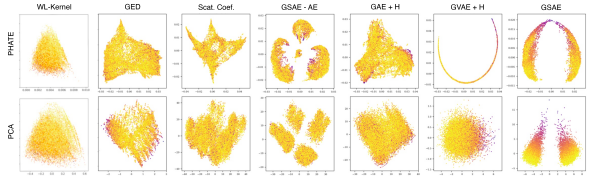

Figure 3. SEQ3 embedding comparison of various embeddings. SEQ3 is known to be bistable (Höbartner and Micura, 2003), with two energy minima which only GSAE reveals.

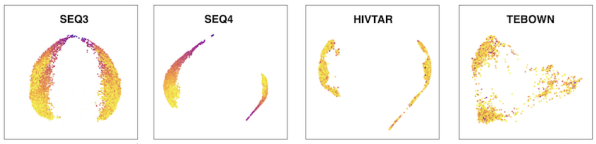

Figure 4. GSAE embeddings of all four RNA sequence structures plotted using PHATE.

5. Results

Toy Data We generate a toy dataset by starting with a randomly generated Erdos-Renyi (ER) graph containing 10 nodes, with edge probability $p = 0.5$. Then for 9999 steps, we randomly chose an edge to remove or add to the previous graph in sequence. This generates a sequence of 10000 graphs that should roughly form single trajectory based on graph edit distance. These toy graphs are visualized in Figure 2A. We visualize these embeddings in two different ways, with PHATE (Moon et al., 2019) a non-linear visualization reduction method that keeps local and global structure, as well as PCA. We see that only the GSAE uncovers the linear trajectory of the graph indicating that simple embedding of edit distances, WL kernels and other graph autoencoders do not uncover the trajectory as well. Further, we quantify the structure in these embeddings in Figure 2B by computing the *graph dirichlet energy* of the signal formed by the sequence index, i.e., the signal $f = [0, 1, \ldots, 10000]$ with Laplacian matrix of each embedding $f^T L f$. Lower values indicate more smoothness. We see in Figure-2B that aside from a direct embedding of the graph edit distance, GSAE has the best smoothness.

RNA fold data In order to generate RNA structural graph data, we start with a particular RNA sequence and use the RNAsubopt program of the ViennaRNA (Lorenz et al., 2011) package to generate $100k$ RNA structures. This program performs dynamic programming to exhaustively sample structures within an energy range and returns an approximate energy for each structure. For the purpose of testing embedding quality we chose four sequences that were identified as having specific structures in literature, SEQ3 (Höbartner and Micura, 2003), SEQ4 (Höbartner and Micura, 2003), HIVTAR (Ganser et al., 2019), and TEBOWN (Cordero and Das, 2015). SEQ3 and SEQ4 reside primarily in one of two bistable structures. TEBOWN was designed to be bistable but was described as a "faulty riboswitch" (Cordero and Das, 2015), displaying 3 or more dominant states. HIVTAR (Ganser et al., 2019) refers to the ensemble generated from the transactivation response element (TAR) RNA of HIV. It has been used as a model system for studying RNA structural dynamics and is one of the few RNAs with single native secondary that dominates. We assess the ability of different models to recover these structures and visualize a smooth energy landscape. We train all neural networks with the same penalties, reconstruction as well as energy regression with hyperparameter $\alpha = 0.5$ unless otherwise noted.

Figure 5. Example trajectory from the PHATE embedding of the GSAE latent space and the corresponding RNA graphs.

Figure 3 contains PHATE and PCA visualizations of SEQ3 embeddings, and shows that only the GSAE model organizes the embeddings by both energy and structure despite using the equally weighted reconstruction and regression penalties. Only the GSAE which recapitulates the bistability of SEQ3 and SEQ4 (Höbartner and Micura, 2003) clearly. Energy smoothness quantified for all four RNA sequences in Table 2 and structural smoothness is shown in Table 1. While we do not have the ground truth for organizing structures, we show smoothness by graph edit distances to both the bistable minima in SEQ3 and SEQ4, with the idea that as structures move away from these minima, they will also increase in energy. Figure 4 shows that GSAE can also shed light on the stability landscape of the four RNA structures. SEQ3 and SEQ4 are bistable, while TEBOWN appears to be tristable. However, our embedding shows that HIVTAR can exist in two different fold structures based on the two structures in the embedding, contrary to what is reported in (Ganser et al., 2019). We also emphasize that the GSAE is a generative model, trained as a VAE, therefore, we can sample trajectories of folds in the landscape as potential paths from high to low energy folds. This is depicted on a sample trajectory in Figure 5.

Table 2. Graph dirichlet energy of molecule free energy signal over K-NN graph of embedding. Here "+ H" refers to the addition of the energy prediction network H

	SEQ3	SEQ4	HIVTAR	TEBOWN
GED	0.409 ± 0.014	0.417 ± 0.031	0.105 ± 0.002	0.729 ± 0.039
Scat. Coeff.	0.345 ± 0.009	0.390 ± 0.007	0.105 ± 0.002	0.649 ± 0.025
GAE	0.331 + 0.008	0.345 + 0.008	**0.101±0.002**	0.556 + 0.014
GAE + H	0.128 + 0.006	0.096 + 0.007	0.102 ± 0.005	0.367 ± 0.010
VGAE	0.485 ± 0.014	0.799 ± 0.018	0.124 ± 0.003	0.547±0.016
VGAE + H	0.345 ± 0.009	0.276 ± 0.007	0.119 ± 0.003	0.546 ± 0.014
WL-kernel	0.636 + 0.048	1.091 + 0.083	0.185 + 0.013	0.559 + 0.033
GSAE - AE	0.209 ± 0.003	0.170 ± 0.002	**0.101 ± 0.001**	0.435 ± 0.008
GSAE (no H)	0.396 ± 0.011	0.444 + 0.007	0.105 ± 0.002	0.506 ± 0.014
GSAE	**0.105 ± 0.006**	**0.081 ± 0.003**	0.109±0.002	**0.352 ± 0.026**

Acknowledgements

This research was partially funded by IVADO (l'institut de valorisation des données) [*G.W.*]; Chan-Zuckerberg Initiative grants 182702 & CZF2019-002440 [*S.K.*]; and NIH grants R01GM135929 & R01GM130847 [*G.W., S.K.*]. The content provided here is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

References

Laura R Ganser, Megan L Kelly, Daniel Herschlag, and Hashim M Al-Hashimi. The roles of structural dynamics in the cellular functions of rnas. *Nature Reviews Molecular Cell Biology*, 20(8):474–489, 2019.

Susan J Schroeder. Challenges and approaches to predicting rna with multiple functional structures. *RNA*, 24(12): 1615–1624, 2018.

Feng Gao, Guy Wolf, and Matthew Hirn. Geometric scattering for graph data analysis. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2122–2131, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/gao19e.html.

Fernando Gama, Alejandro Ribeiro, and Joan Bruna. Stability of graph scattering transforms. In *Advances in Neural Information Processing Systems*, pages 8036–8046, 2019a.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016a.

Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(77):2539–2561, 2011.

Fernando Gama, Alejandro Ribeiro, and Joan Bruna. Diffusion scattering transforms on graphs. In *International Conference on Learning Representations*, 2019b.

Ronald R. Coifman and Mauro Maggioni. Diffusion wavelets. *Applied and Computational Harmonic Analysis*, 21(1):53 – 94, 2006.

Stéphane Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.

Michael Perlmutter, Feng Gao, Guy Wolf, and Matthew Hirn. Understanding graph neural networks with asymmetric geometric scattering transforms. *arXiv preprint arXiv:1911.06253*, 2019.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Claudia Höbartner and Ronald Micura. Bistable secondary structures of small rnas and their structural probing by comparative imino proton nmr spectroscopy. *Journal of molecular biology*, 325(3):421–431, 2003.

Kevin R Moon, David van Dijk, Zheng Wang, Scott Gigante, Daniel B Burkhardt, William S Chen, Kristina Yim, Antonia van den Elzen, Matthew J Hirn, Ronald R Coifman, et al. Visualizing structure and transitions in high-dimensional biological data. *Nature Biotechnology*, 37(12):1482–1492, 2019.

Ronny Lorenz, Stephan H Bernhart, Christian Höner Zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. Viennarna package 2.0. *Algorithms for molecular biology*, 6(1):26, 2011.

Pablo Cordero and Rhiju Das. Rich rna structure landscapes revealed by mutate-and-map analysis. *PLoS computational biology*, 11(11), 2015.

Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017.

Daniel C Elton, Zois Boukouvalas, Mark D Fuge, and Peter W Chung. Deep learning for molecular design—a review of the state of the art. *Molecular Systems Design & Engineering*, 4(4):828–849, 2019.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Peter Steffen, Björn Voß, Marc Rehmsmeier, Jens Reeder, and Robert Giegerich. Rnashapes: an integrated rna analysis package based on abstract shapes. *Bioinformatics*, 22(4):500–503, 2006.

Luan Lin, Wilson H McKerrow, Bryce Richards, Chukiat Phonsom, and Charles E Lawrence. Characterization and visualization of rna secondary structure boltzmann

ensemble via information theory. *BMC bioinformatics*, 19(1):82, 2018.

Nathan A Siegfried, Steven Busan, Greggory M Rice, Julie AE Nelson, and Kevin M Weeks. Rna motif discovery by shape and mutational profiling (shape-map). *Nature methods*, 11(9):959–965, 2014.

Aleksandar Spasic, Sarah M Assmann, Philip C Bevilacqua, and David H Mathews. Modeling rna secondary structure folding ensembles using shape mapping data. *Nucleic acids research*, 46(1):314–323, 2018.

Chanin T Woods, Lela Lackey, Benfeard Williams, Nikolay V Dokholyan, David Gotz, and Alain Laederach. Comparative visualization of the rna suboptimal conformational ensemble in vivo. *Biophysical journal*, 113(2): 290–301, 2017.

Zichao Yan, William Leif Hamilton, and Mathieu Daniel Blanchette. Graph neural representational learning of rna secondary structures for predicting rna-protein interactions. *bioRxiv*, 2020.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016b.

Miloš Daković, Ljubiša Stanković, and Ervin Sejdić. Local smoothness of graph signals. *Mathematical Problems in Engineering*, 2019, 2019.

A. Ablation Study

We train three variations of the GSAE itself. First we examine the effect of the variational formulation by including results for GSAE-AE (our model trained as a vanilla autoencoder). We also truncate the regressor network H of GSAE, which we refer to as GSAE (no reg). Lastly, we compare and show GSAE also improves upon simply embedding geometric scattering coefficients, which may contain information for organizing the graphs, but are not selected, weighted or combined as well as in the proposed GSAE.

B. Related Work on Graph Embeddings

Graph edit distances (GED) are a way of measuring the distances between graphs based on the number of elementary operations needed to change from one graph to another. These elementary operations involve vertex insertions and deletions, edge insertions and deletions, etc. Distances can directly be embedded using MDS or indirectly via a Gaussian kernel using a kernel-PCA method such as diffusion maps (Coifman and Lafon, 2006) or the more recently proposed PHATE (Moon et al., 2019) which collects manifold information for visualization in two dimensions. Another approach to embedding a graph is the Weisfeiler-Lehman (WL) kernel (Shervashidze et al., 2011) which maps a graph to a sequence of graph that encapsulate graph topological features.

Graph neural networks have been used primarily for classifying nodes. However, methods such as graph variational autoencoders (GVAEs) (Kipf and Welling, 2016a) can be used for embedding nodes. However, in order to achieve invariance, node embeddings have to be pooled. Typically, similar to convolutional neural networks, graph neural networks are pooled using sum or max pooling (Hamilton et al., 2017). Here, inspired by deep scattering transforms (Gao et al., 2019), we instead use the statistical moments of node activations for pooling.

There has been much work in recent years using graph-based methods for a related class of biomolecules, commonly referred to in the literature as *small molecules*. For a review of these methods we refer the reader to (Elton et al., 2019). In this related domain, a similar approach to organizing biomolecules in latent space using an auxiliary loss has been previously studied by (Gómez-Bombarelli et al., 2018). However their approach relies on RNNs to encode and decode a domain-specific string representation for this class of biomolecules.

In regards to RNA secondary structures, few experiment-free, graph-based approaches to interpreting a RNA secondary structure folding ensemble (a collection of folds arising from a single sequence) have been studied. The popular RNAShapes software seeks to abstract the structural diversity of a folding ensemble in a coarser set of possible graphs (Steffen et al., 2006). MIBPS is a method which utilizes mutual information between folds of an ensemble to predict multi-modality (Lin et al., 2018). Furthermore, several works rely on chemical probing data in order to infer multi-modality in folding ensembles (Cordero and Das, 2015) (Siegfried et al., 2014),(Spasic et al., 2018), (Woods et al., 2017). Closer to the deep learning literature is a recent work by (Yan et al., 2020) where they train a GNN-based model to study RNA secondary structures in the context of RNA binding proteins (RBPs).

C. Model Implementation Details

C.1. GSAE

In this work we begin with graphs G on which we place diracs to use as node signals. We then generate a set of node features using the scattering transform formulation depicted in Figure 1C and described in Section 4. To achieve a graph representation, we summarize node features using statistical moments from (Gao et al., 2019) rather than the traditional sum or max operation We refer to this graph representation S.

The GSAE model takes as input the summarized scattering coefficients, S. In the GSAE model, shown in Figure 1A, we use 2 fully-connected layers with RELU activations followed by the reparameterization operation described in (Kipf and Welling, 2016a). Batchnorm layers from (Ioffe and Szegedy, 2015) are interspersed between the initial encoding layers. The decoder of GSAE is comprised of 2 fully-connected layers with a RELU activation function on the non-output layer. For the regressor network, we an identical module as the decoder, only differing the size of the output layer. The loss which is optimized during training becomes,

$$L = L_{recon} + \alpha L_{pred} + \beta L_{D_{KL}}$$

or,

$$L = \frac{1}{N}||\hat{\Phi}, \Phi||_2^2 + \alpha \frac{1}{N}||\hat{y} - y||_2^2 + \beta D_{\mathrm{KL}}\left(q(z|\Phi)\|p(z)\right)$$

Training runs consisted of 15000 iterations using a batch size of 100. We used PyTorch's Adam optimizer with a learning rate of 0.0001. For experimental results presented in Table 1 and Table 2, we use a bottleneck dimension of 25.

C.2. Scatting Inverse Network Model

From the GSAE, we are able to produce a latent space where both information about graph structure and graph

metaproperties are preserved. However the GSAE construction differs from other graph autoencoders as it reconstructs summarized scattering coefficients rather than graphs. This presents an obstacle when generating graphs from points in the latent space. We remedy this by training an additional model referred to as the Scattering Inverse Network (SIN) model.

Similar to GSAE, SIN uses an autoencoder architecture which reconstructs scattering coefficients. However SIN differs from GSAE as it produces the graph adjacency matrix in it's middle latent representation. This endows SIN with the capacity to effectively invert scattering coefficients and consequently, allow for generation of graphs from the GSAE's latent space.

For SIN, depicted in Figure 1B, we use 2 blocks of *fully-connected layer, RELU , batchnorm* followed by a final fully-connected layer. This final fully-connected layer expands the representation so that the inner-product decoder of GAE (Kipf and Welling, 2016a) may be applied to produce an adjacency matrix representation of the graphs. Unique to SIN is that we then convert the adjacency matrix to scattering coefficients $\hat{S}$ using the original scattering cascade used to construct the input to GSAE.

We train SIN by first pre-training the scattering inverse module which takes S to $\hat{A}$ using a binary-cross entropy loss. Once this loss has converged, we then refine the generator by training on the overall reconstruction of S. We show these final MSE losses for the RNA datasets in Table 3.

Table 3. Inverse model test set reconstruction error generating adjacency matrices from scattering coefficients over N=10 runs

	MSE $\pm$ std $\times 10^{-3}$
SEQ3	0.070 ± 0.010
SEQ4	0.059 ± 0.004
HIVTAR	7.425 ± 2.459
TEBOWN	7.175 ± 3.552

C.3. GAE and GVAE

For our comparisons to traditional graph autoencoder formulations, we compare against the GAE and GVAE from (Kipf and Welling, 2016a). Though more complex graph autoencoders have been developed for domain-specific applications (e.g. small molecules from chemistry), we focus on a more general sense of graph embeddings which do not rely on existing node features but rather only utilize graph structure and an associated meta-property.

To make set-up as similar to GSAE as possible, we again begin with featureless graphs G on which we place diracs as the initial node signal. The GAE and GVAE both use this initial signal to create meaningful node features using graph convolutional (GCN) layers from (Kipf and Welling, 2016b). In this work we use 2 GCN layers with RELU activation functions for both GAE and GVAE.We then attain a graph-wise representation using the same pooling as GSAE, which uses the first 4 statistical moments across the node dimension. The resulting vector is then passed through two fully-connected layers to produce the final latent representation which is used for evaluations. We train these models using a binary-cross entropy loss for 15000 iterations with batch size set to 100. As with GSAE, we use PyTorch's Adam optimizer with a learning rate of 0.0001.

D. Embedding space interpolation

The inverse model described in Section C.2 can be used in a generative setting to produce sequences of graphs that resemble RNA folding trajectories. To achieve this we first train a GSAE model with small latent space dimension over RNA graphs from one of the datasets. Then for two randomly chosen RNA graphs in the dataset we sample from the line segment connecting their corresponding latent space embeddings. These interpolated points in the latent space are mapped into the space of scattering coefficients by the decoding network of the GSAE. Finally these points in scattering coefficient space are fed into the inverse model SIN. The weights of the resulting adjacency matrices are rounded to produce unweighted graphs.

To see this method in action we trained the GSAE model with latent space dimension 5 on 70,000 graphs from the SEQ3 dataset. In selecting the end points for our generative trajectories, we sampled the starting graph from the subset of high-energy configurations and the final graph from the low-energy configurations. See Figure 6 for trajectories generated using this method. In Figure 7 for every trajectory we compute the graph edit distance between the final graph and each individual graph in the trajectory. The results suggest that in most cases, these generative trajectories are smooth in terms of graph edit distance.

E. Energy Prediction

We show the energy prediction accuracy of the models at various settings of the parameter α in Table 4 which decides the penalty balance between the autoencoding reconstruction penalty and the energy prediction penalty. We see that the GSAE is able to simultaneously organize the embedding structurally and predict a metaproperty of the graphs successfully.

F. Smoothness Metric

In this work, we quantify the smoothness of a signal in embedding space using graph dirichlet energy. This metric can

Table 4. Performance of auxiliary network H. Energy prediction MSE (mean ± std. over 10 runs) on each of the four RNA datasets

	SEQ3	SEQ 4	HIVTAR	TEBOWN
GAE	224.832 ± 291.277	360.797 ± 416.404	217.451 ± 190.157	168.191 ± 205.224
GAE + H ($\alpha = 0.1$)	1.223 ± 0.069	1.364 ± 0.119	3.159 ± 0.090	0.624 ± 0.031
GAE + H ($\alpha = 0.5$)	1.247 ± 0.0084	1.377 ± 0.101	3.174 ± 0.078	0.608 ± 0.025
VGAE	99.442 ± 7.386	156.922 ± 10.508	207.148 ± 12.742	10.028 ± 2.431
VGAE + H ($\alpha = 0.1$)	5.536 ± 0.089	6.996 ± 0.234	3.168 ± 0.045	0.741 ± 0.021
VGAE + H ($\alpha = 0.5$)	4.338 ± 0.0789	5.625 ± 0.434	3.188 ± 0.037	0.750 ± 0.015
GSAE - AE	2.875 ± 0.04	3.877 ± 0.053	3.176 ± 0.044	0.678 ± 0.01
GSAE (no H)	98.561 ± 3.35	156.567 ± 4.292	209.654 ± 8.425	8.930 ± 2.948
GSAE ($\alpha = 0.1$)	1.786 ± 0.639	2.908 ± 0.788	3.739 ± 0.477	0.722 ± 0.008
GSAE ($\alpha = 0.5$)	1.795 ± 0.533	2.040 ± 0.587	3.509 ± 0.201	0.661 ± 0.246

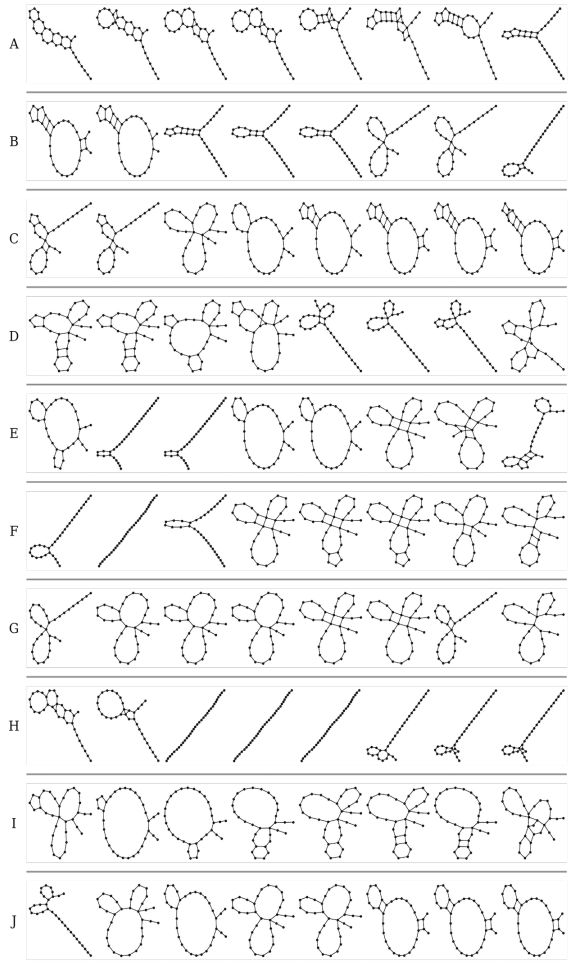

Figure 6. Sample trajectories produced by applying the scattering inverse network to linear interpolations between training points in GSAE latent space.

be interpreted as the squared differences between neighboring nodes which should be small if the signal is smooth and slow varying across latent space. Conversely, large differences in the quantity of interest between neighboring nodes would produce as large value of this metric. Here we use a normalized form of the graph dirchlet energy, described in (Daković et al., 2019) as a smoothness index, which takes the form,

$$\lambda_x = \frac{\mathbf{x}^T \mathbf{L} \mathbf{x}}{\mathbf{x}^T \mathbf{x}}$$

The graph dirchlet energy requires that we first form a graph on our embeddings in order to compute the graph Laplacian $\mathbf{L}$. We do this using a symmetric k-nearest neighbor (kNN) graph where a data points x_i and x_j are connected by an unweighted edge in the graph if either x_i or x_j fall within each other's kNN. Nearest neighbers are determined using Euclidean distance between points in latent space.

G. Datasets

G.1. Toy Dataset

For evaluation of our model on a noise-less toy dataset, we create a graph trajectory starting from an initial Erdős-Rényi or binomial graph with p=0.5. A step in this trajectory is either an edge addition or deletion. Starting from the initial graph, we take 9999 steps and save each step's graph. After the final step, we have produced a sequence of graphs which we refer to as a trajectory.

G.2. RNA Datasets

The four datasets used in this work were generated using ViennaRNA's RNAsubopt program. This program takes as input an RNA sequence and produces a set of folds. Here we used the "-e" option which produces an exhaustive set of folds within a specified kcal/mol energy range above the minimum free energy (MFE) structure. We then split each dataset into a train and test split with a ratio of 70:30.

- **SEQ3**: SEQ3 is an artificial RNA sequence of 32 nu-

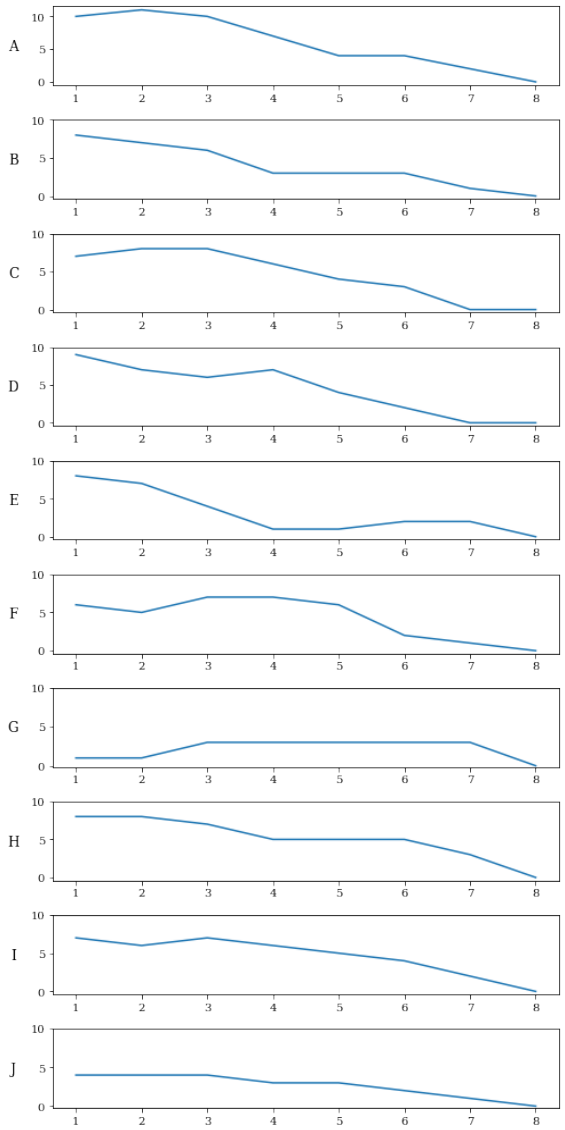

Figure 7. GED to end points. Each row corresponds to trajectories presented in Figure 6. A decrease in GED between the i-th and j-th step demonstrates that the step j's graph is a more similar closer to the graph at the end point in terms of GED

cleotides designed to be bistable (Höbartner and Micura, 2003). We use an energy window of 25kcal/mol which produces a total of 472859 sequences. We then reduce this set to 100k structures by sampling without replacement.

- **SEQ4**: SEQ4 is also an artificial RNA sequence of 32 nucleotides and is bistable (Höbartner and Micura, 2003). We use a 30kcal/mol window which produces 926756 structures. We then reduce this set to 100k structures by sampling without replacement.
- **HIVTAR**: HIVTAR is 61 nucleotides long and from the literature (Ganser et al., 2019), is expected to be monostable. We use a 22kcal/mol window which produces 1529527 structures. We then reduce this set to 100k structures by sampling without replacement.
- **TEBOWN**: TEBOWN has a sequence length of 72 nucleotides and is expected to be multistable (Cordero and Das, 2015). We use a 9kcal/mol window which produces 151176 structures. We then reduce this set to 100k structures by sampling without replacement.

H. GSAE Embedding Quality

H.1. Nearest Neighbor Experiments

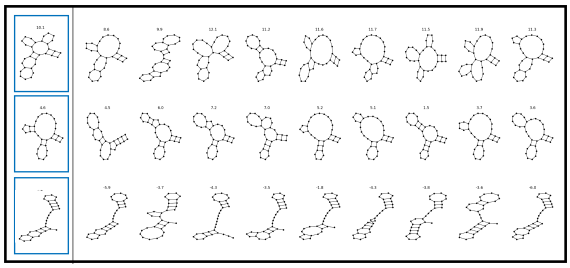

Figure 8. 3 samples from SEQ3 and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

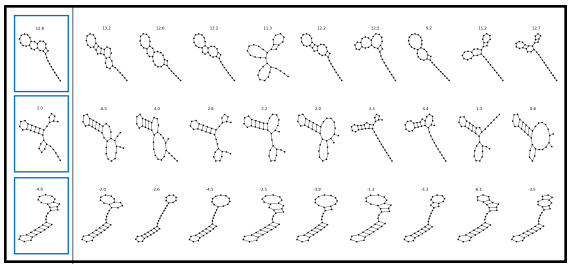

Figure 9. 3 samples from SEQ4 and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

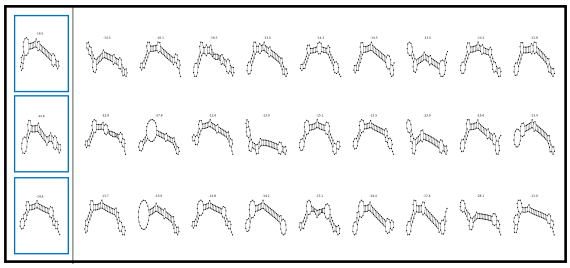

Figure 10. 3 samples from HIVTAR and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

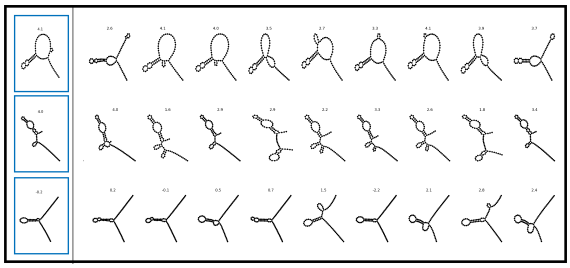

Figure 11. 3 samples from TEBOWN and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

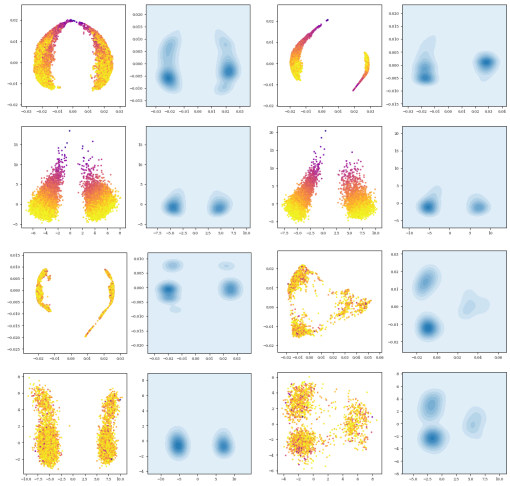

Figure 12. Density plots from PHATE and PCA coordinates of RNA embeddings. 25-dimensional embeddings are generated using GSAE and are plotted using PHATE and PCA . The density plot is shown to the right of it's corresponding PHATE and PCA plot. *Top row*: SEQ3, SEQ4. *Bottom Row*: HIVTAR, TEBOWN.

H.2. Density Plots

Here in Figure 12 we show the density plots for the PHATE and PCA plots of GSAE embeddings for each of the four RNA datasets. As described in the paper, we recapitulate the bistable nature of SEQ3 and SEQ4 and visualize this further in the Figure 12 (top row). In the HIVTAR dataset, we view two clusters of structures rather than the expected single cluster. We hypothesize that this separation may be a result of a minor structural distinction due to the low variability between structures in the HIVTAR dataset. Lastly, we also show that the TEBOWN dataset displays > 2 minima in its density plots (bottom right), which is expected in (Cordero and Das, 2015). Notably, as the energy increases and grows further away from that of the minimum free energy structure, the number of structures possible increases. As a result, instable and structurally diverse folds make up a large portion of RNA folding ensembles.