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

Egbert Castro
Comp. Biol. and Bioinf. Program
Yale University
New Haven, CT 06511
egbert.castro@yale.edu

Andrew Benz
Dept. of Mathematics
Yale University
New Haven, CT 06511
andrew.benz@yale.edu

Alexander Tong
Dept. of Computer Science
Yale University
New Haven, CT 06511
alexander.tong@yale.edu

Guy Wolf
Mila – the Québec AI institute
Dept. of Mathematics and Statistics
Université de Montréal
Montréal, QC H3T 1J4
guy.wolf@umontreal.ca

Smita Krishnaswamy
Depts. of Genetics and Computer Science
Yale University
New Haven, CT 06520
smita.krishnaswamy@yale.edu

Abstract

Biomolecular graph analysis has recently gained much attention in the emerging field of geometric deep learning. While numerous approaches aim to train classifiers that accurately predict molecular properties from graphs that encode their structure, an equally important task is to organize 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. Our approach is based on the intuition that geometric scattering generates multi-resolution features with in-built invariance to deformations, but as they are unsupervised, these features may not be tuned for optimally capturing relevant domain-specific properties. We demonstrate the effectiveness of our approach to data exploration of RNA foldings. Like proteins, RNA molecules can fold to create low energy functional structures such as hairpins, but the landscape of possible folds and fold sequences are not well visualized by existing methods. We show that GSAE organizes RNA graphs both by structure and energy, accurately reflecting bistable RNA structures. Furthermore, it enables interpolation of embedded molecule sequences mimicking folding trajectories. Finally, using an auxiliary inverse-scattering model, we demonstrate our ability to generate synthetic RNA graphs along the trajectory thus providing hypothetical folding sequences for further analysis.

1 Introduction

An emerging focus in deep learning is the ability to analyze graph structured data. Historically, these types of data have mainly originated from network analysis fields, such as the study of social networks or citation networks. [1–3]. More recently, interest in graph data analysis has also risen in the equally important study of *biomolecules*, represented by graphs that model their molecular structure. For this

Preprint. Under review.

work, we define *biomolecules* as chemical and biological entities that form structure though atomic interactions, such as small-molecules, proteins, and RNA. This data domain presents both great challenge and opportunity. The challenge arises from the (1) the discrete nature of graph structure data, (2) the vast space of possible structures possible ($10^{30} - 10^{60}$ drug-like molecules for example [4]), and (3) the underlying physical constraints for validity. However, advancements in this domain have the potential for greater insights into biological questions as well as improvements in drug discovery.

While RNA is sometimes thought of as a linear sequence of bases, non-coding RNA especially can itself fold into a structure that has functionality [5]. 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 uncover features of their folding landscape, as well as reflect their transitions smoothly. This goal can motivate the examination 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, in order to change functionality, i.e. whether is a riboswitch–a question which is difficult to answer normally.

Our purpose for generating such an embedding is three-fold: First, obtaining faithful embeddings where neighbors are close in both in terms of graph structure and in terms of molecular properties. Second, enabling visual exploration and interpretation of biomolecular structures using the embedding space. Third, the ability to sample trajectories of folds in the embedding space and decode them into molecular folds.

Here we propose a new framework for organizing biomolecular structures called geometric scattering autoencoder (GSAE). GSAE first encodes a graph based on. the recently proposed graph scattering transform [6, 7] coefficients of dirac signals placed on the nodes of the molecular graph. Scattering transforms are essentially deep, multiscale wavelet transforms of signals [8] whose coefficients are summarized into statistical moments to guarantee invariance to perturbation. Scattering transforms were recently translated into the graph domain by way of graph signals (node features), which we employ here by placing dirac signals on the nodes of RNA graphs. Next GSAE uses several feedforward layers to further refine and organize the scattering coefficients into a meaningful embedding layer based both on the reconstruction penalty (typical of an autoencoder) and auxiliary penalties to predict molecular properties. To generate graphs we train an additional autoencoder, a scattering inversion network (SIN) that takes scattering coefficients as inputs and generates adjacency matrices in the embedding layer.

We show that together the GSAE and SIN can be used for faithful embedding and visualization spaces of RNA structures as well as synthetic graphs, as well as generation and interpolation of molecular fold trajectories.

We compare our results to several of the most prominent GNN-based graph representation approaches including GAE [9], GVAE [9], as well as non-trainable methods like embeddings of the WL-kernel computed on graphs [10] or embeddings of graph edit distance matrices. We note that the GSAE both qualitatively and quantitatively produces the best organized embeddings. Most importantly, we compare to directly embedding geometric scattering coefficients [6, 7]. GSAE also improves vastly upon simply embedding geometric scattering coefficients, which may contain information to organize the graphs but not selected, weighted or combined as well as with the GSAE neural network.

The ability of the GSAE to reorganize graph datasets is first demonstrated on a toy dataset representing a random graph trajectory. We then compare on four datasets that are generated by graph folding software ViennaRNA [11] on RNA sequences that are known to have bistable, and multistable structures.

2 Background

RNA Secondary Structures: Though historically thought of as simply an intermediate between DNA and an encoded protein, recent findings have reversed this long-held paradigm and instead point to a cell where various RNA are major drivers of processes, both aberrant and healthy [5]. A consequence of this new understanding is an increased focus on connecting RNA structure and function.

Difficulty in experimentally determining the structure of RNA molecules [12] has drawn efforts towards computational approaches to RNA secondary structure prediction. Dynamic programming algorithms, such as Zuker's [13] or Nussinov's algorithm [14], that search for the minimum free energy (MFE) structure are the most popular approach to predicting the secondary structure. However, this focus on the MFE draws attention away from structures whose energies may be slightly above that of the MFE and may hinder new findings for two reasons. The first of which is that the dynamic and crowded intracellular environment is likely to prevent RNA from folding into its MFE [15]. Secondly, inaccuracies in the energy function used by the RNA secondary structure prediction method may produce a false MFE. Notably, the Zuker algorithm's accuracy drops as sequences grow in length. For these reasons it is vital for one to examine the larger set of folds possible to fully understand the structural diversity of a given sequence, such as subset shown for SEQ3 in Figure 5.

RNA exists in its lowest order structure as a chain of monomers, namely nucleotides, which are able to interact with each other to form intra-chain hydrogen bonds. The collection of these interaction can be interpreted as edges between the nucleotides, which are consequently nodes. In this way, we can view the secondary structure of RNA as a graphs. This viewpoint allows for the application of existing tools from both graph signal processing and graph neural networks to be leveraged in this domain.

Geometric Scattering: While the adjacency matrix contains the total information of the connectivity of a graph, it also enables graph representation learning by using tools from graph signal processing. These tools extract features that encode meaningful and readily usable information about graph structures and variations between them, which in turn produce whole-graph embedding in Euclidean spaces that can be used for further analysis. Here, we utilize the recently presented geometric scattering transform [6, 16] for this purpose. This transform is based on a cascade of graph wavelets, typically constructed via diffusion wavelets [17] based on the diffusion operator from [18] .

While the diffusion map framework [18] was originally presented in the context of manifold learning, it can be adapted to graph settings by constructing 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 where D is a diagonal matrix of its vertex degrees. Then, for any $t > 0$, the matrix P^t contains t-step diffusion transition probabilities between graph nodes. On the other hand, these powers of P can also be interpreted as lowpass filters that average signals over multiscale diffusion neighborhoods in the graph, 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 [17] combines these lowpass and highpass filters (which are considered here on graphs) 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 the 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. [6] propose to follow the same approach as in expected scattering of traditional signals [8, 19] to aggregate diffusion wavelet coefficients by taking statistical moments after applying nonlinearity in form of absolute value. Namely, they compute the first-order scattering features as

$$S_1 f = [\| \, |\mathbf{\Psi}_j f| \, \|_q]_{1 \leq j \leq J, 1 \leq q \leq Q} \tag{1}$$

These first order scattering coefficients capture the statistics of signal variations over the graph. They are complimented on one hand by zeroth-order scattering, consisting simply of the statistical moments of the signal f itself (without filtering), and on the other hand with higher order scattering coefficients that capture richer variations that are eliminated by the aggregation in Eq. 1. In general, the m^{th} order scattering features are computed by a cascade of m wavelet transforms and nonlinear absolute value operations, 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 the 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 the signal f. We refer the reader to [6, 20, 7, 16] for further details on geometric scattering and its properties.

Autoencoders: The ability of unsupervised deep learning methods to recover structure in data has been realized in recent years in a variety of domains, including biology. Much of this work has been done through the use of the autoencoder model [21], which optimizes a reconstruction objective. This architecture can be used to filter information from data of dimension d that is not important to reconstruction, and thus can be assumed to be less informative, by setting the number of neurons n in the middle layers of the model such that $n < p$. The final p-dimensional representation can be extracted after training and can be used for downstream tasks such as classification or, as we do here, visualization. Recently, the concept of autoencoding has been extended to graphs with the advent of graph autoencoders [9] and graph variational autoencoders [9].

3 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 [18] or the more recently proposed PHATE [22] which collects manifold information for visualization in two dimensions. Another approach to embedding a graph is the Weisfeiler-Lehman (WL) kernel [10] 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) [9] 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 [23]. Here, inspired by deep scattering transforms [6], we instead use the statistical moments of node activations for pooling.

4 Methods

Notations: Let $G = (V, E, W)$ denote a graph with *n* vertices $V = \{v_1, \ldots, v_n\}$, edges $E \subseteq \{(v_\ell, v_m) : 1 \leq \ell, m \leq n\}$, and weights $W = \{w\,(v_\ell, v_m) > 0 : (v_\ell, v_m) \in E\}$. Let A denote the $n \times n$ weighted adjacency matrix of the graph defined by $A_{i,j} = w(v_i, v_j)$ for $(v_i, v_j) \in E$, and zero otherwise.

4.1 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. 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 and 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 two 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

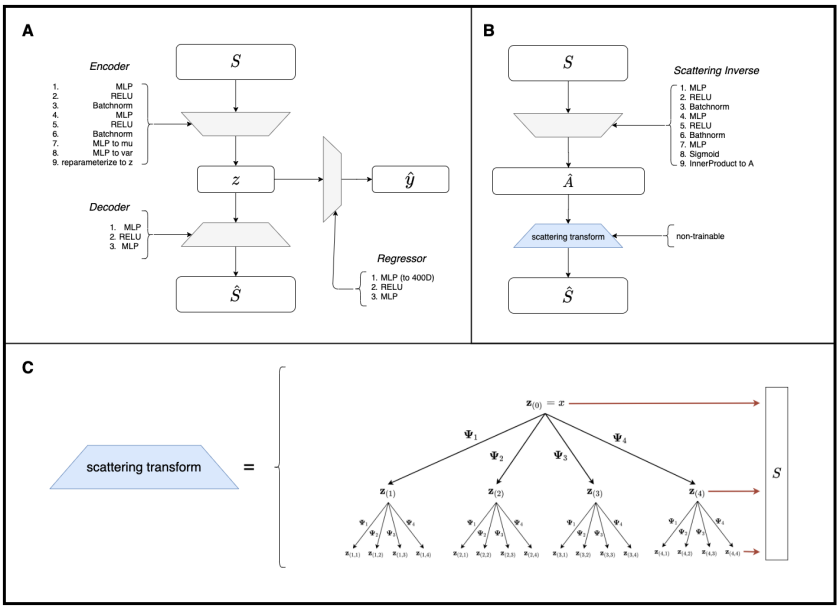

Figure 1: A. GSAE, B. Inverse transform, C. Scattering transform network.

that can be difficult to compute from the graph. In principle, this smoothness could be enforced for multiple meta properties.

4.2 The Geometric Scattering AutoEncoder Model

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. In the following we describe the operation of our embedding method on a single graph, but its training (and utilization for the analyses in the following sections) rely on the application of these steps to each of the graphs in the data.

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 may assume that there is node correspondence between graphs, and thus we can produce a set of dirac signals $d_i = \{0, \ldots, 1, \ldots, 0\}$ that provide one-hot encoding of each of the nodes v_i in the graph (i.e., $d_i[j] = 1$ iff $i = j$, and zero otherwise). As they do not encode any relation between nodes, these signals can be considered equivalent across all graphs, or rather edge structures, instantiated over the same set of nodes.

Next, we map an input graph to a Euclidean feature space given by the scattering features of these dirac signals over 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 [6], 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. 4.1, 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 distribution over the data to resemble normal distribution via KL divergence. We refer the reader to [24] for further details and derivation of such terms in VAEs.

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.

5 Results

Here we compare the GSAE model to several trained and untrained models including, graph edit distance embeddings, WL-kernel embeddings [10], direct embedding of graph scattering coefficients [6], graph autoencoders (GAE), graph variational autoencoder (GSAE) [9], and two variations of the GSAE itself, a GSAE-AE (our model trained as a vanilla autoencoder), and GSAE (trained as a variational autoencoder). Note that this constitutes an ablation study as we have tested, just using scattering coefficients (leaving the neural network off), using graph neural networks directly on data graphs, and training without the variational penalties. We show using both visualizations of the embedding as well as quantifications that GSAE better organizes RNA graphs both by structure and energy. For visualizations we set $\alpha = 0.5$ which weights reconstruction and energy regression equally. Network architecture and parameters used are specified in the supplement. We also examine scattering inverse model's ability to generate folds, located in Sec. B.

Table 1: Graph dirichlet energy of sequence position signal for random graph dataset.

Method	k = 5	k = 10
GED	0.0015	0.003
Scat. Coeff.	0.447	1.723
WL-kernel	2.647	5.319
AE	0.618	1.262
GAE	0.108	0.342
GVAE	0.451	0.853
GSAE	0.0579	0.129

Toy Data In order to explore the biomolecular embedding properties of the GSAE framework, we first compare the model on a toy dataset. The toy dataset is generated 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 2. We visualize these embeddings in two different ways, with PHATE [22] 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 Table 1 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 Table 1 that aside from a direct embedding of the graph edit distance, GSAE has the best smoothness. However, note that in general graph edit distances are very expensive to compute and cannot be trained to predict meta-properties as we desire in biomolecules.

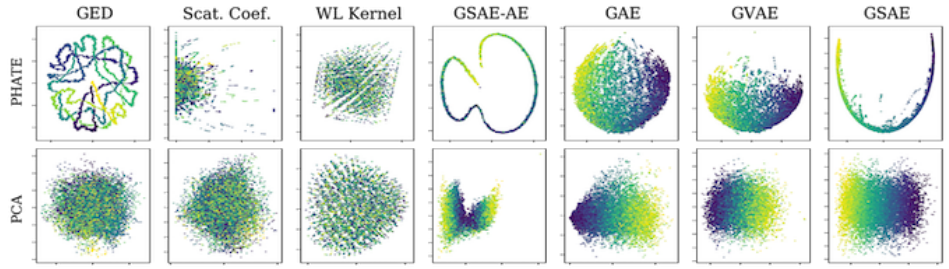

Figure 2: PHATE and PCA plots of seven different embeddings of the random graph dataset. Color corresponds to the position in the 10,000-step sequence of graphs, the ordering of which GSAE reveals clearly.

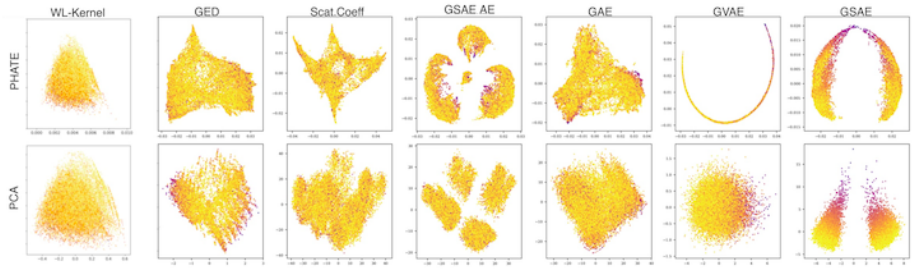

Figure 3: SEQ3 embedding comparison of various embeddings, trained embeddings have $\alpha = 0.5$. SEQ3 is known to be bistable [25], with two energy minima which only GSAE reveals.

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 [11] 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 purposes of testing embedding quality we chose four sequences that were identified as having specific structures in literature, SEQ3 [25], SEQ4 [25], HIVTAR [5], and TEBOWN [26]. SEQ3 and SEQ4 contain RNA hairpin structures and reside primarily in one of two bistable structures. Bistability is typical of riboswitches, or RNA molecules whose functionality can turn "ON" or "OFF" based on how they are folded. TEBOWN was designed to be bistable but was described as a "faulty riboswitch" [26]. Instead of being bistable it has 3 or more states. HIVTAR [5] refers to the ensemble for 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 for which single native secondary is thought to dominate.

Figure 3 shows the embeddings of the various models using PHATE and PCA again. Here we see 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 shows the bistable structure [25] clearly. We believe that the other models lose information pertaining to the entire graph

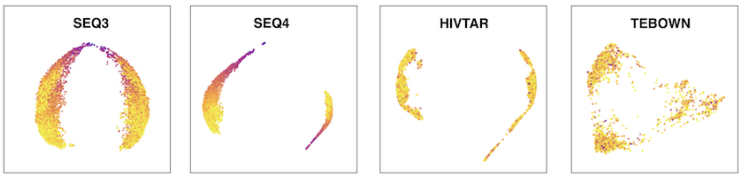

Figure 4: GSAE embeddings of all four RNA sequence structures plotted using PHATE with energy regression penalty hyperparameter $\alpha = 0.5$. We confirm the known bistability of SEQ3 and SEQ4 [25]. We hypothesize three dominant structures for TEBOWN [26] based on the embedding, and two dominant structures for HIVTAR.

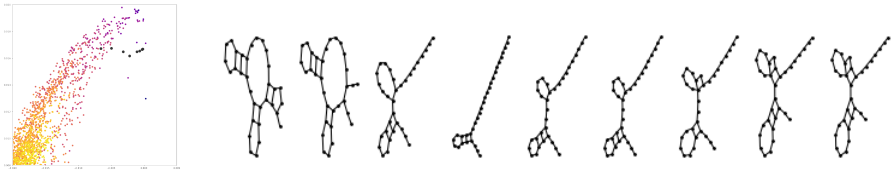

Figure 5: Example trajectory from the PHATE embedding of the GSAE latent space and the corresponding RNA graphs. Additional interpolation results shown in Sec. B

Table 2: Results show structural organization of the various embeddings on the two bistable datasets. Graph dirichlet energy with the graph edit distance from the two stable energy minima are reported.

	SEQ3				SEQ4			
	Min 1		Min 2		Min1		Min 2	
	K=5	K=10	K=5	K=10	K=5	K=10	K=5	K=10
GED	0.442 ± 0.0003	0.999 ± 0.0006	0.517 ± 0.0002	0.116 ± 0.0006	0.045 ± 0.0003	0.101 ± 0.0006	0.058 ± 0.0003	0.129 ± 0.0007
Scat. Coeff.	0.0604 ± 0.0003	0.1408 ± 0.0007	0.0732 ± 0.0002	0.172 ± 0.0003	0.066 ± 0.0002	0.152 ± 0.0004	0.0859 ± 0.0005	0.198 ± 0.0011
GAE	0.035 ± 0.001	0.079 ± 0.002	0.045 ± 0.002	0.101 ± 0.004	0.038 ± 0.001	0.085 ± 0.002	0.053 ± 0.003	0.117 ± 0.006
GAE + reg	0.044 ± 0.006	0.098 ± 0.014	0.06 ± 0.006	0.13 ± 0.012	0.043 ± 0.003	0.094 ± 0.008	0.062 ± 0.003	0.134 ± 0.007
VGAE	0.425 ± 0.006	0.84 ± 0.012	0.478 ± 0.008	0.944 ± 0.017	0.443 ± 0.007	0.876 ± 0.015	0.528 ± 0.008	1.045 ± 0.015
VGAE + reg	0.392 ± 0.005	0.772 ± 0.01	0.46 ± 0.008	0.905 ± 0.015	0.405 ± 0.006	0.792 ± 0.01	0.469 ± 0.006	0.917 ± 0.012
WL-Kernel	0.185 ± 0.0012	0.423 ± 0.0022	0.225 ± 0.001	0.514 ± 0.0017	0.2 ± 0.0016	0.456 ± 0.002	0.263 ± 0.0016	0.56 ± 0.004
GSAE - AE	0.069 ± 0.001	0.157 ± 0.002	0.087 ± 0.002	0.195 ± 0.003	0.069 ± 0.001	0.151 ± 0.002	0.085 ± 0.002	0.186 ± 0.003
GSAE	0.337 ± 0.021	0.657 ± 0.050	0.381 ± 0.027	0.743 ± 0.064	0.112 ± 0.004	0.246 ± 0.007	0.038 ± 0.001	0.298 ± 0.007
GSAE + reg	0.346 ± 0.076	0.691 ± 0.148	0.402 ± 0.074	0.798 ± 0.143	0.103 ± 0.004	0.229 ± 0.009	0.124 ± 0.005	0.274 ± 0.011

structure, unlike the scattering coefficients. The GSAE-AE (trained as an autoencoder) seems to create additional cluster seperations without the KL-divergence penalty to create embedding cohesion. Energy smoothness quantified for all four RNA sequences in Table 3 and structural smoothness is shown in Table 2. 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, i.e., structures close to the minima must have similar folds.

Figure 4 shows that GSAE can also shed light on the stability landscape of the four RNA structures. GSAE embeddings indeed show that SEQ3 an SEQ4 are bistable, while TEBOWN looks 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 [5], explored in the supplement.

Table 3: Graph dirichlet energy of molecule free energy signal over K-NN graph of embedding.

	SEQ3		SEQ4		HIVTAR		TEBOWN	
Method	K=5	K=10	K=5	K=10	K=5	K=10	K=5	K=10
GED	0.409 ± 0.014	0.822 ± 0.028	0.417 ± 0.031	0.873 ± 0.069	0.105 ± 0.002	0.208 ± 0.003	0.729 ± 0.039	1.390 ± 0.069
Scat. Coeff.	0.345 ± 0.009	0.699 ± 0.018	0.390 ± 0.007	0.828 ± 0.017	0.105 ± 0.002	0.206 ± 0.004	0.649 ± 0.025	1.269 ± 0.048
GAE	0.331 + 0.008	0.666 + 0.014	0.345 + 0.008	0.717 + 0.017	**0.101±0.002**	0.199±0.004	0.556 + 0.014	1.078 + 0.028
GAE + reg	0.128 + 0.006	0.272 + 0.014	0.096 + 0.007	0.206 + 0.016	0.102 ± 0.005	0.202 ± 0.004	0.367 ± 0.010	0.741 ± 0.020
VGAE	0.485 ± 0.014	0.960 ± 0.021	0.799 ± 0.018	1.579 ± 0.035	0.124 ± 0.003	0.245 ± 0.005	0.547±0.016	1.079 ± 0.032
VGAE + reg	0.345 ± 0.009	0.680 ± 0.016	0.276 ± 0.007	0.542 ± 0.014	0.119 ± 0.003	0.237 ± 0.006	0.546 ± 0.014	1.082 ± 0.030
WL-kernel	0.636 + 0.048	1.244 + 0.078	1.091 + 0.083	2.063 + 0.115	0.185 + 0.013	0.353 + 0.017	0.559 + 0.033	1.141 + 0.048
GSAE - AE	0.209 ± 0.003	0.429 ± 0.006	0.170 ± 0.002	0.354 ± 0.005	**0.101 ± 0.001**	**0.197 ± 0.003**	0.435 ± 0.008	0.859 ± 0.014
GSAE	0.396 ± 0.011	0.795 + 0.022	0.444 + 0.007	0.922 + 0.016	0.105 ± 0.002	0.207 ± 0.004	0.506 ± 0.014	0.994 ± 0.028
GSAE + reg	**0.105 ± 0.006**	**0.219 ± 0.012**	**0.081 ± 0.003**	**0.169 + 0.007**	0.109±0.002	0.214 ± 0.005	**0.352 ± 0.026**	**0.710 ± 0.050**

Table 4: Energy prediction MSE (mean ± std. over 10 runs) on each of the four RNA sequences.

	SEQ3	SEQ 4	HIVTAR	TEBOWN
GAE	224.832 ± 291.277	360.797 ± 416.404	217.451 ± 190.157	168.191 ± 205.224
GAE ($\alpha = 0.1$)	1.223 ± 0.069	1.364 ± 0.119	3.159 ± 0.090	0.624 ± 0.031
GAE ($\alpha = 0.5$)	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 ($\alpha = 0.1$)	5.536 ± 0.089	6.996 ± 0.234	3.168 ± 0.045	0.741 ± 0.021
VGAE ($\alpha = 0.5$)	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	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

We show the energy prediction accuracy of the models at various settings of the parameter α 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. 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.

6 Conclusion

We have presented a hybrid approach that combines geometric scattering transforms on graphs with variational autoencoders in order to embed molecular graphs. We propose this as a useful way of learning a large multi-scale set of features that are descriptive of graph structure, and then using an autoencoder to create an embedding by selecting and combining these features appropriately. We demonstrate the utility of this in the RNA secondary structure domain, to identify the energy minima and folding landscapes of functional RNA, and to generate folding trajectories.

Broader Impact

This work and provides tools to explore physical and chemical properties of biomolecules in general which are of growing interest in the fields of machine learning and data science especially with the recent rise of geometric deep learning and intensive work on graph representative learning. This work has widespread applicability in learning representations of entire graphs rather than nodes, and as a result can be used in situations where large sets of graphs need to be ordered and explored including in medicine (drug molecules), protein design, gene regulatory networks of different graphs, etc. We do not believe this research puts any group at a disadvantage or has potential for adverse use, beyond any other algorithmic embedding techniques.

References

[1] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[2] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

[3] Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo. Semi-supervised learning with graph learning-convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11313–11320, 2019.

[4] Pavel G Polishchuk, Timur I Madzhidov, and Alexandre Varnek. Estimation of the size of drug-like chemical space based on gdb-17 data. *Journal of computer-aided molecular design*, 27(8):675–679, 2013.

[5] 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.

[6] 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`.

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

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

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

[10] 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.

[11] 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.

[12] Hao Zhang, Chunhe Zhang, Zhi Li, Cong Li, Xu Wei, Borui Zhang, and Yuanning Liu. A new method of rna secondary structure prediction based on convolutional neural network and dynamic programming. *Frontiers in genetics*, 10, 2019.

[13] Michael Zuker and Patrick Stiegler. Optimal computer folding of large rna sequences using thermodynamics and auxiliary information. *Nucleic acids research*, 9(1):133–148, 1981.

[14] Ruth Nussinov, George Pieczenik, Jerrold R Griggs, and Daniel J Kleitman. Algorithms for loop matchings. *SIAM Journal on Applied mathematics*, 35(1):68–82, 1978.

[15] Quan Zou, Mao-zu GUO, and Tao-tao ZHANG. A review of rna secondary structure prediction algorithms. *Acta Electronica Sinica*, 36(2):331, 2008.

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

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

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

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

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

[21] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[22] 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.

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

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

[25] 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.

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

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

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

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

A Model Implementation Details

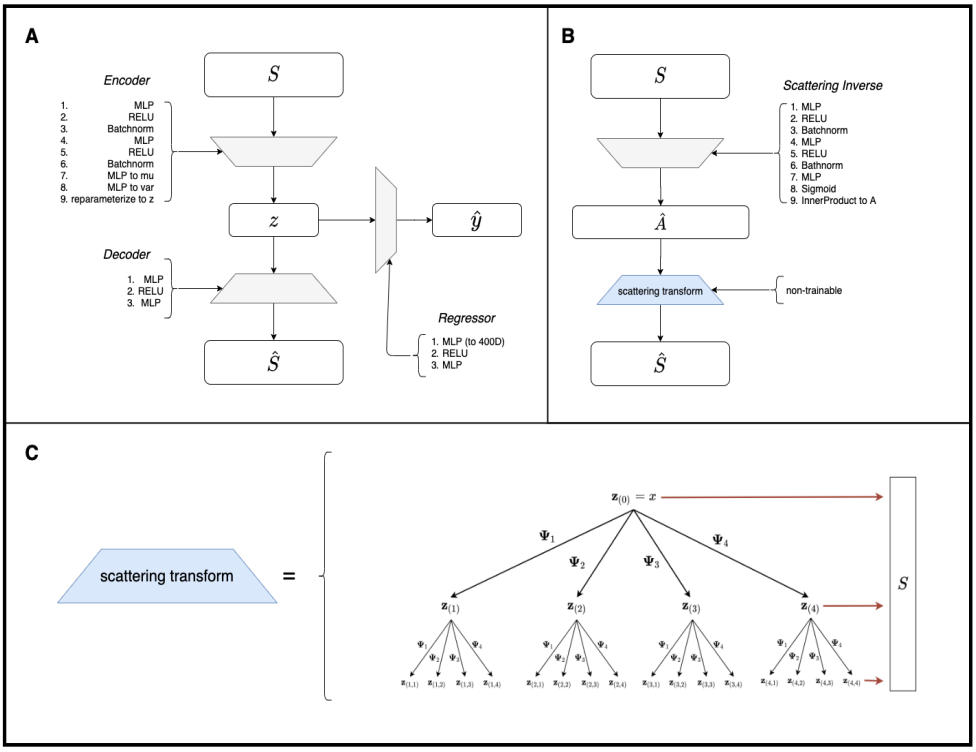

Figure 6: Model architecture

A.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 6C and described in Section 2. To achieve a graph representation, we summarize node features using statistical moments from [6] 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 6A, we use 2 fully-connected layers with RELU activations followed by the reparameterization operation described in [9]. Batchnorm layers from [27] 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,

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

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 3 and Table 2, we use a bottleneck dimension of 25.

A.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 6B, 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 [9] 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 5.

Table 5: 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

A.3 GAE and GVAE

For our comparisons to traditional graph autoencoder formulations, we compare against the GAE and GVAE from [9]. 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 [28]. 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.

B Embedding space interpolation

The inverse model described in Section A.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 7 for trajectories generated using this method. In Figure 8 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.

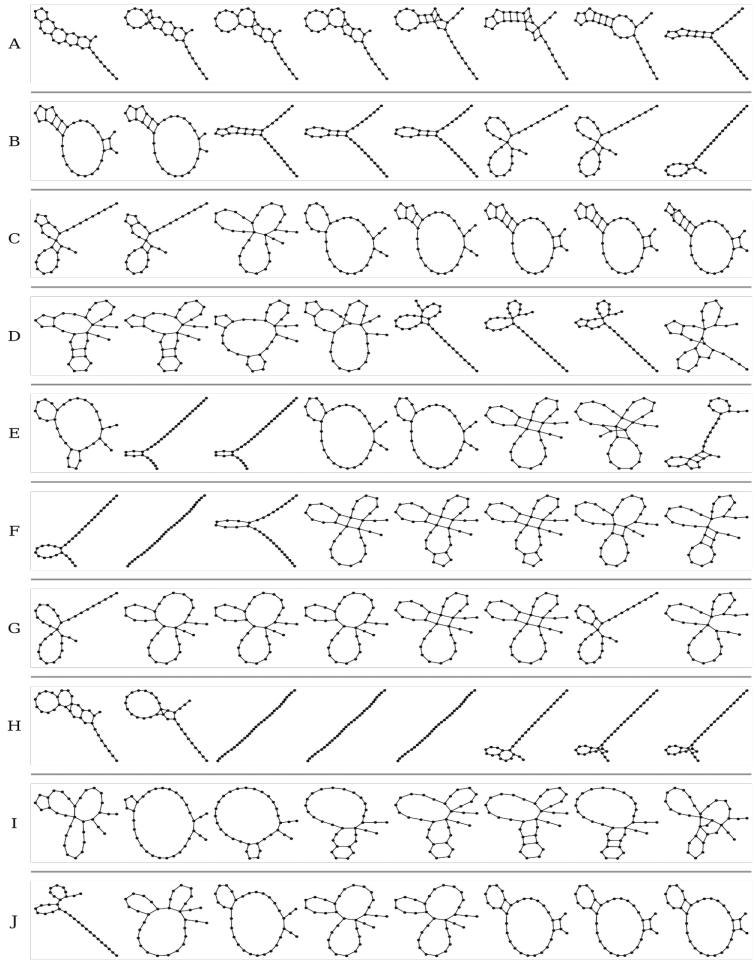

Figure 7: Sample trajectories produced by applying the scattering inverse network to linear interpolations between training points in GSAE latent space.

C Smoothness Metric

In this work, we quantify the smoothness of a signal in embedding space using graph dirichlet energy. This metric can 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 [29] as a smoothness index, which takes the form,

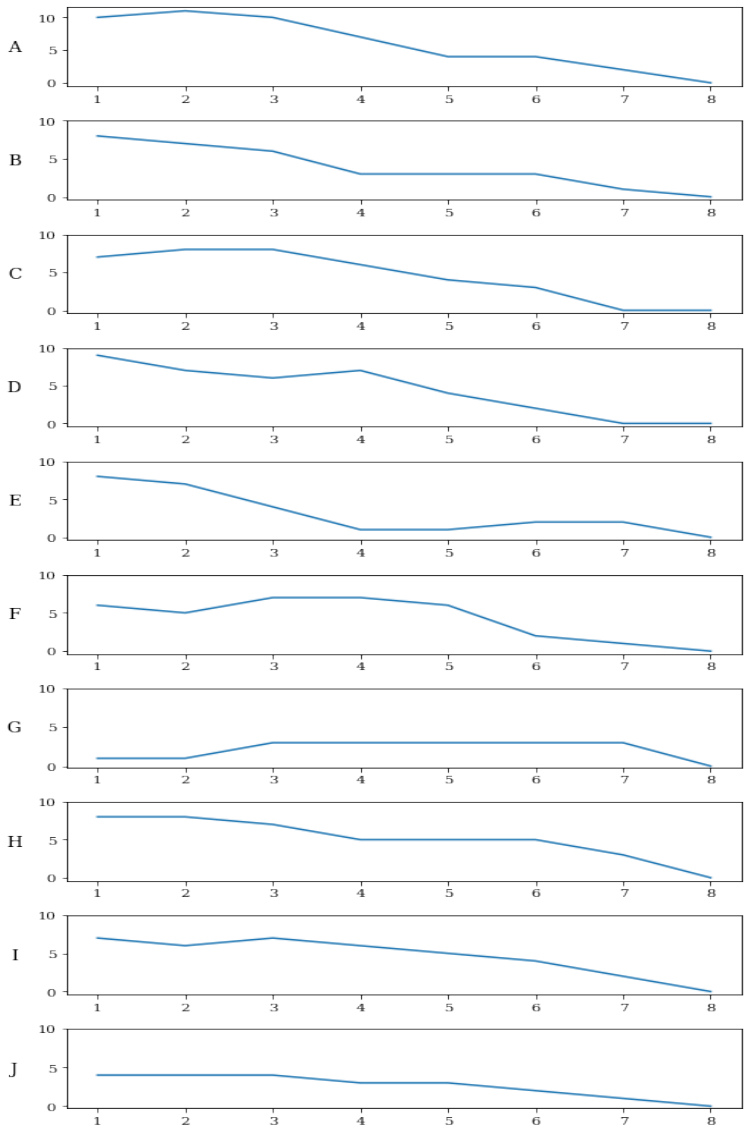

Figure 8: Edit distance between each individual graph in the trajectory and the final graph in the trajectory for all the trajectories in Figure 7.

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

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 edge in the graph if either x_i or x_j fall within each other's kNN.

D Datasets

D.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.

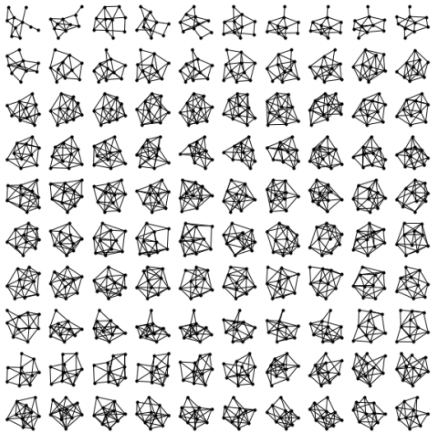

Figure 9: First 100 steps in the toy graph trajectory

D.2 RNA Datasets

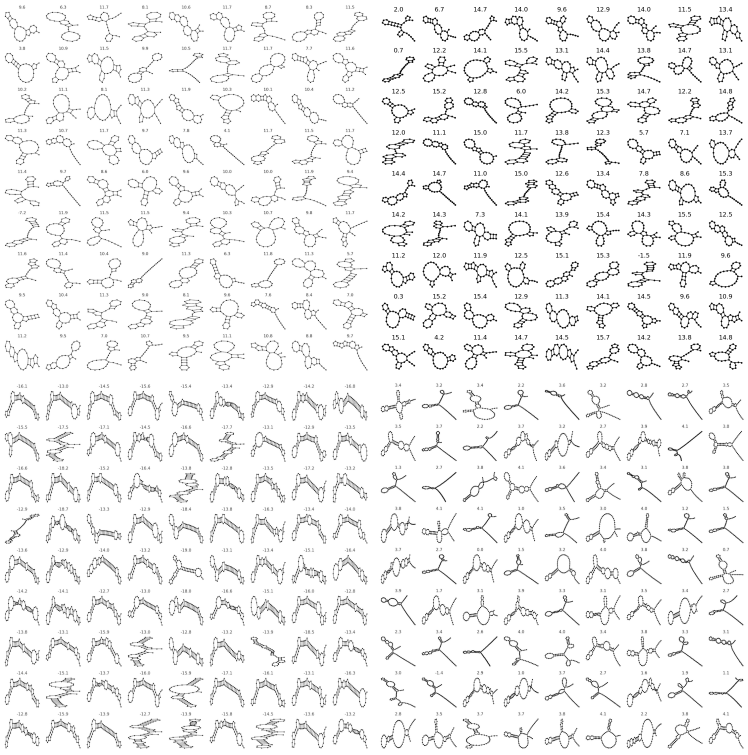

Figure 10: Samples from RNA datasets. 100 samples plotted from each RNA dataset. *Top Row*: SEQ3, SEQ4. *Bottom Row*: HIVTAR, TEBOWN. Values are each structure's energy (kcal/mol)

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 nucleotides designed to be bistable [25]. 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 [25]. 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 [5], 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 [26]. We use a 9kcal/mol window which produces 151176 structures. We then reduce this set to 100k structures by sampling without replacement.

E GSAE Embedding Quality

E.1 Nearest Neighbor Experiments

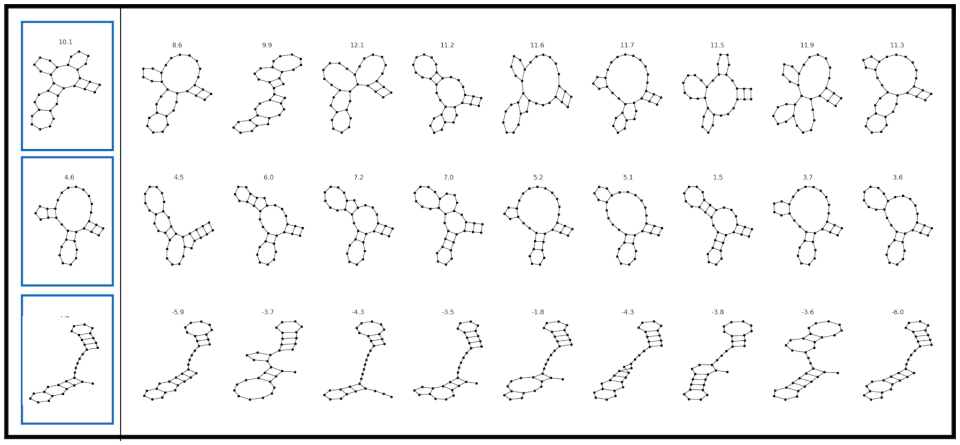

Figure 11: 3 samples from SEQ3 and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

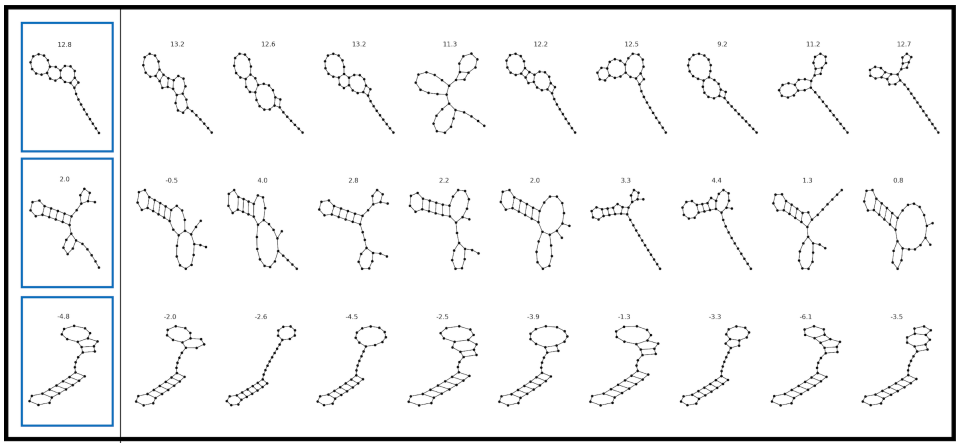

Figure 12: 3 samples from SEQ4 and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

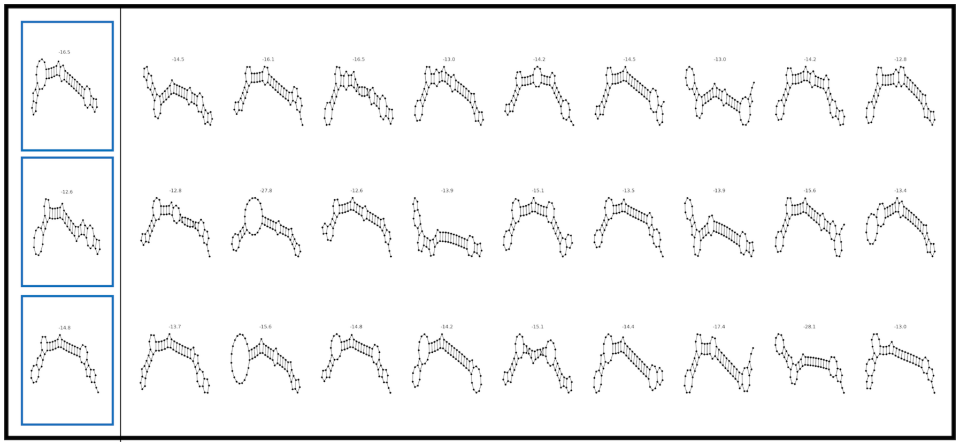

Figure 13: 3 samples from HIVTAR and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

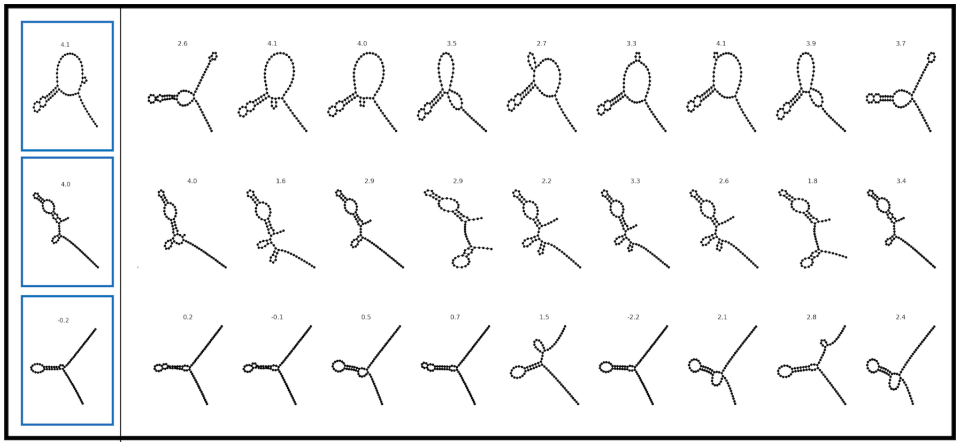

Figure 14: 3 samples from TEBOWN and their 9 nearest neighbors in GSAE latent space. Values are each structure's energy (kcal/mol)

E.2 Density Plots

Here in Figure 15 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 15 (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, observable in Figure 10 (bottom left). Lastly, we also show that the TEBOWN dataset displays >2 minima in its density plots (bottom right), which is expected in [26]. 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.

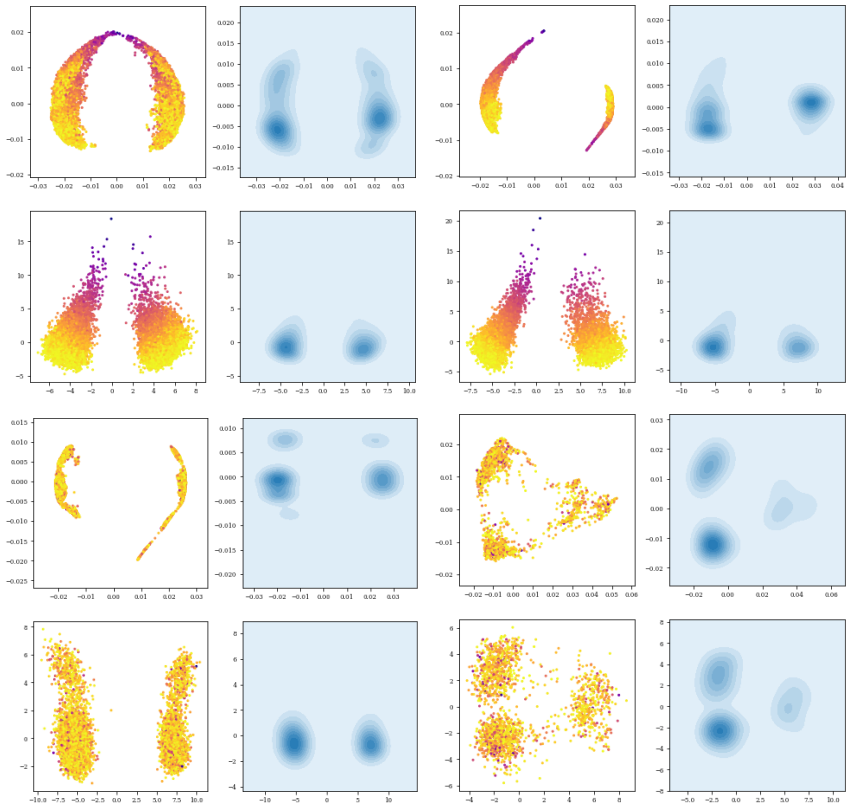

Figure 15: Density plots from PHATE and PCA coordinates of RNA embeddings. 25-dimensional embeddings are generated using GSAE and are plotted using PHATE and PCA . The density plot is shown to the right of it's corresponding PHATE and PCA plot. *Top row*: SEQ3, SEQ4. *Bottom Row*: HIVTAR, TEBOWN.