PROGAE: A GEOMETRIC AUTOENCODER-BASED GENERATIVE MODEL FOR DISENTANGLING PROTEIN DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding molecular dynamics is critical, especially for large flexible molecules like proteins. Protein function, as well as modulations thereof due to ligand binding or changes in environment, are intimately connected with these dynamics. This work focuses on learning a generative neural network on molecular simulation trajectories to characterize the distinct dynamics of a protein bound to various drug molecules. Specifically, we use a geometric autoencoder framework to learn separate latent space encodings of the intrinsic and extrinsic geometries of the system. For this purpose, the proposed Protein Geometric AutoEncoder (ProGAE) model is trained on the length of the alpha-carbon pseudobonds and the orientation of the backbone bonds of the protein. Using ProGAE latent embeddings, we reconstruct protein simulation trajectories at or near the experimental resolution. Empowered by the disentangled latent space learning, the extrinsic latent embedding is successfully used for classification or property prediction of different drugs bound to a specific protein. Additionally, ProGAE is able to be transferred to a trajectory of a different state of the same protein or to a completely different protein, where only the dense layer decoding from the latent representation needs to be retrained. Results show that our geometric learning-based method enjoys both accuracy and efficiency for generating complex molecular dynamics, charting the path toward scalable and improved approaches for analyzing and enhancing molecular simulations.

1 INTRODUCTION

The complex and time-consuming calculations in molecular simulations have been significantly impacted by the application of machine learning techniques in recent years. In particular, deep learning has been applied to analysis and simulation of molecular trajectories to address diverse problems, such as estimating free energy surfaces, defining optimal reaction coordinates, constructing Markov State Models, and enhancing molecular sampling. For a comprehensive review of deep learning methods for analyzing and enhancing molecular simulations, see (Noé et al., 2020a) and (Noé et al., 2020b).

Specifically, there has been interest in modeling the underlying conformational space of protein dynamics via deterministic (Ramaswamy et al., 2020) or probabilistic (Varolgüneş et al., 2020; Bhowmik et al., 2018; Guo et al., 2020) autoencoder (AE) -based approaches. This line of work has mainly attempted to respect the domain geometry by using convolutional AEs on features extracted from 3D structures. In parallel, learning directly from 3D structure has recently developed into an exciting and promising application area for deep learning. In this work, we learn the protein dynamics directly from the protein structures using geometric deep learning. We also investigate how the geometry of the proteins itself can assist learning and improving latent conformational space interpretability. To this end, we propose a Protein Geometric AutoEncoder model, named ProGAE, to separately encode intrinsic and extrinsic protein geometries.

The main contributions of this work are summarized:

- Inspired by the recent unsupervised geometric disentanglement learning works (Tatro et al., 2020; Wu et al., 2019; Yang et al., 2020), we propose a novel geometric autoencoder named ProGAE that

directly learns from protein 3D structures by separately encoding the intrinsic and the extrinsic geometries into disjoint latent spaces and generates protein structures from these latent spaces.

- We further propose a novel formulation, in which network intrinsic input is taken as the C_α-C_α pseudo-bond distances, and the extrinsic input is the backbone bond orientations.
- Analysis shows that the learned extrinsic geometric latent space can be used for drug classification and drug property prediction, where the drug is bound to the given protein.
- Experiments confirm that the extrinsic geometry of the proteins generated by ProGAE is essentially invariant to changes in the intrinsic latent code. This allows for better latent space interpretability.
- We also demonstrate that the learned ProGAE can be transferred to a trajectory of the protein in a different state or a trajectory of a different protein all-together.

1.1 Related Work

Recently, there has been a body of work on using deep learning to learn from structures involving proteins (Graves et al., 2020; Klicpera et al., 2020; Jing et al., 2020). For example, Gainza et al. (2019) uses geometric deep learning to predict docking sites for protein interactions. Ingraham et al. (2019a) solves the *inverse folding problem* using a graph transformer on the protein backbone. Degiacomi (2019) uses an AE to generate candidate proteins for docking. Hermosilla et al. (2020) leverages the notion of intrinsic and extrinsic geometry to define an architecture for fold classification task.

Additionally, there has been focus on learning molecular dynamics from simulation trajectories. Wang et al. (2020) learns the dynamics between a particular starting conformation and a particular target conformation by training a control Hamiltonian represented by a graph neural network. From the perspective of Hamiltonian systems, Chen et al. (2020) introduces Symplectic RNNs which leverages sympletic integration when learning the dynamics of a physical system. Ingraham et al. (2019b) learns dynamics via a differentiable simulator using Langevin dynamics. Mardt et al. (2018) introduces VAMPnet which maps molecular coordinates to Markov states capturing molecular kinetics. Lee et al. (2019) augments molecular dynamics simulation with deep learning to improve the sampling of the folded states of proteins. Tsai et al. (2020) uses a LSTM language model to predict protein dynamics. Additionally, several other works exist that use AE-based approaches for analyzing the protein conformational space (Ramaswamy et al., 2020; Varolgüneş et al., 2020; Bhowmik et al., 2018; Guo et al., 2020).

Another line of work involves enhanced sampling with neural net approximations of slow variables (Chen & Ferguson, 2018; Chen et al., 2019; Ribeiro et al., 2018). Deep and adversarial learning have been employed for enhanced sampling (Bonati et al., 2019; Zhang et al., 2019). Noé et al. (2019) uses deep generative neural net to directly sample the equilibrium distribution of a many-body system defined by an energy function, without using molecular simulation.

These mentioned works have not considered explicit disentangling of intrinsic and extrinsic geometries. To the best of our knowledge, this work is the first to propose an autoencoder for the unsupervised geometric disentanglement of protein conformational space captured in molecular dynamics simulations. This disentangled representations provide better interpretability of the latent space in terms of the physico-chemical and geometric attributes.

2 ProGAE for Protein Dynamics

First, we introduce a novel geometric autoencoder (ProGAE) for generating protein dynamics. After that, We discuss the input signals, and how they can reconstruct the protein coordinates.

Geometric Features of Protein (Pseudo)Bonds as Network Input ProGAE functions by separately encoding intrinsic and extrinsic geometry with the goal of achieving better latent space interpretability. To be precise, we here clarify the notions of intrinsic and extrinsic geometry. Mathematically, we can consider a manifold independently of its embedding in Euclidean space. Properties that do not depend on this Euclidean embedding are said to be intrinsic geometric properties, while properties that do are referred to be extrinsic. As an example, given two atoms of a protein, the

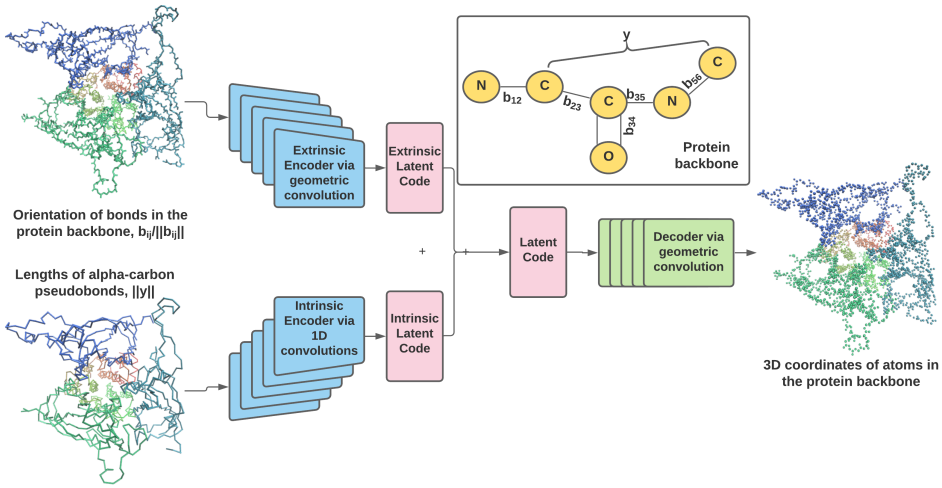


Figure 1: Architecture of geometric autoencoder, ProGAE, that we train to generate protein dynamics by separate encodeing of information related to coarse intrinsic and extrinsic geometries. These geometries are captured via the orientation of bonds in the backbone (extrinsic) and length of $C_\alpha - C_\alpha$ pseudobonds (intrinsic). Concatenation of these latent representations is used to generate the 3D coordinates of the backbone atoms.

intrinsic distance between them is the minimum sum of bond lengths in the bond path connecting them, whereas the extrinsic distance is just the Euclidean distance between them in $\mathbb{R}^3$. For a more in-depth discussion of intrinsic and extrinsic geometry, we refer the reader to (Do Carmo, 2016).

We note that the protein backbone trace can be viewed as a discretized curve in Euclidean space. The protein backbone itself can be viewed as a discretized curve locally. Both objects are depicted in Figure 1 with their geometric features as network input. To be clear, a discretized curve is a curve approximated by a polygonal chain. Without difficulty, we can see that a polygonal chain can be determined up to translation given both the length and orientation of its line segments. Then it follows that the protein backbone can be determined given the length and orientation of its bonds. Here the length of these bonds is intrinsic while the orientation is extrinsic. Thus, to decouple the intrinsic and extrinsic geometry, we can consider encoding these signals.

While we could use the length of the bonds in the backbone as the intrinsic signal input into the network, this is not the signal that we use. This is because the length of covalent bonds undergoes very little change during a simulation performed using an empirical force-field, like the simulations considered in this work. A standard deviation of less than 0.059Å from target bond lengths is common in PDB structures (Jaskolski et al., 2007). To this end, we consider intrinsic geometry at a coarser level, so that the resulting signal has more variability. Specifically, we consider the length of the C_α-C_α pseudobonds in the backbone trace as a representative of the intrinsic geometry of a protein, where as backbond bond orientations are considered for capturing extrinsic geometry.

We model the backbone by the graph, $\mathcal{G}_b = (\mathbb{V}_b, \boldsymbol{E}_b)$, and the backbone trace by the graph, $\mathcal{G}_t = (\mathbb{V}_t, \boldsymbol{E}_t)$. Then our intrinsic and extrinsic signals, $Int : \boldsymbol{E}_t \to \mathbb{R}$ and $Ext : \boldsymbol{E}_b \to \mathbb{R}^3$ are defined:

$$Int(E_{ij}) = \|E_{ij}\|_2, \quad E_{ij} \in \boldsymbol{E}_t, \qquad Ext(E_{ij}) = sgn(j-i)\frac{E_{ij}}{\|E_{ij}\|}, \quad E_{ij} \in \boldsymbol{E}_b. \tag{1}$$

Network Architecture With the network inputs defined, we discuss the architecture of ProGAE. The core idea is to create an *intrinsic* latent space, $L_I \in \mathbb{R}^{n_i}$, and an *extrinsic* latent space, $L_E \in \mathbb{R}^{n_e}$, via separately encoding the intrinsic and extrinsic signals. To this end, our network contains two encoders, Enc_i and Enc_e where:

$$Enc_i \circ Int(\boldsymbol{E}_t) \in L_I, \qquad Enc_e \circ Ext(\boldsymbol{E}_b) \in L_E. \tag{2}$$

We then jointly decode these latent vectors to recover the coordinates of the atoms in the protein backbone. Thus, we formally define the decoder:

$$Dec : L_I \times L_E \to \mathbb{R}^{|\mathbb{V}_b| \times 3}. \tag{3}$$

This high level structure of ProGAE is depicted in Figure 1. We now describe the architecture of the encoders and decoders. As these edge-based signals are defined on a geometric domain, it is sensible to learn feature representations using geometric convolution that respects the geometry of the data. The intrinsic encoder is the most simple, as the signal is defined on the backbone trace, which corresponds to a set of discrete curves. Here each curve corresponds to a protein fragment. Then we define Enc_i to be a series of 1D convolutions operating on each protein fragment. Each convolution is taken to have a stride of 2, and is followed with batch normalization layers and ReLU.

In contrast, the extrinsic encoder operates on a signal defined on the protein backbone, which we associate with a graph. So the graph attention layers of graph attention networks (GATs) introduced in (Veličković et al., 2017) are a natural tool to use in our architecture, albeit with some modification. Since the input signal is defined only on the edges of the graph, $\boldsymbol{E}_b$, we define a signal on the graph vertices, $\mathbb{V}_b$, as the average value of its incident edges. Then the first layer of the extrinsic encoder uses the edge-convolution operator of (Gong & Cheng, 2019) to map this graph signal to a signal defined exclusively on the graph vertices, $\mathbb{V}_b$. The rest of the encoder contains successive graph attention layers with sparsity defined by a given neighborhood radius and a stride of 2. Given L layers, this can be thought to define a sequence of graphs, $\{\mathcal{G}_{b,i}\}_{i=0}^{L}$, with increasing decimation. As with Enc_i, each layer is followed with batch normalization and ReLU.

Global average pooling is applied to the output of each encoder to introduce invariance to the number of vertices in $\mathbb{V}_t$ and $\mathbb{V}_b$. Dense layers then map each result to their respective latent spaces, L_I and L_E. The Tanh activation function is applied to bound the latent space. This produces the intrinsic and extrinsic latent codes, $\boldsymbol{z}_i$ and $\boldsymbol{z}_e$.

To decode from the latent embedding, the latent code $\boldsymbol{z}$ is taken to be the concatenation of the two latent codes, $[\boldsymbol{z}_i, \boldsymbol{z}_e]$. A dense layer then maps $\boldsymbol{z}$ to the a signal defined on the most decimated backbone graph, $\mathcal{G}_{b,L}$. The structure of the decoder, Dec, is then analogous to Enc_e, though the convolutions are essentially transposed. The output of Dec is the point cloud, $\hat{\boldsymbol{P}}$, corresponding to the predicted coordinates of the backbone atoms, $\mathbb{V}_b \approx \boldsymbol{P}$.

Loss Function The first term in the loss function is a basic reconstruction loss, where $\boldsymbol{P}$ and $\hat{\boldsymbol{P}}$ are taken to be the true and predicted coordinates of the protein backbone atoms. Namely, we evaluate their difference using Smooth-L_1 loss. This loss is defined as

$$SmoothL_1(\boldsymbol{x}, \boldsymbol{y}) := \sum_{i=1}^{\#\boldsymbol{x}} z_i, \quad \text{where} \quad z_i = \min\left(\frac{\delta^2}{2}(x_i - y_i)^2, \delta|x_i - y_i| - \frac{1}{2}\right), \tag{4}$$

where we choose $\delta = 2$. This loss function can roughly be viewed as L_2 loss altered to be more robust to outliers and prevent exploding gradients (Girshick, 2015).

As the reconstruction loss depends on the embedding of the protein in Euclidean space, it may not necessarily best measure if the intrinsic geometry is being faithfully reconstructed. We would also like to promote the function of the intrinsic latent space during training. To be clear, we consider two encoded proteins with latent codes, $[\boldsymbol{z}_{i,1}, \boldsymbol{z}_{e,1}]$ and $[\boldsymbol{z}_{i,2}, \boldsymbol{z}_{e,2}]$. Then we form a new latent variable,

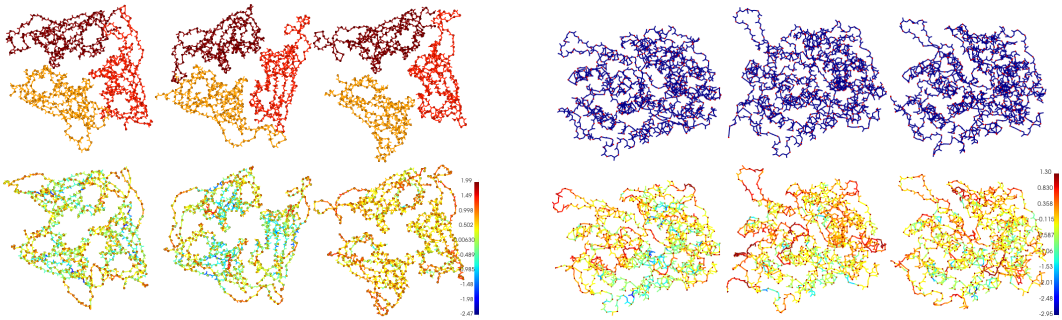
$$\hat{\boldsymbol{z}}_i = (1-\beta)\boldsymbol{z}_{i,1} + \beta\boldsymbol{z}_{i,2}, \quad \hat{\boldsymbol{z}}_e = \boldsymbol{z}_{e,1}, \qquad \beta \sim \mathbf{U}[0,1]. \tag{5}$$

Each of these latent variable decodes to some point cloud $\hat{\boldsymbol{P}}$. We let $Int(\boldsymbol{E}_{t,\beta})$, $Int(\boldsymbol{E}_{t,1})$, and $Int(\boldsymbol{E}_{t,2})$ be the lengths of the C_α-C_α pseudobonds of the generated proteins from the interpolated latent code and the two given latent codes. We then introduce a bond length penalty given by,

$$\mathcal{R}(\hat{\boldsymbol{P}}_1, \hat{\boldsymbol{P}}_2; \beta) = ||Int(\boldsymbol{E}_{t,\beta}) - ((1-\beta)Int(\boldsymbol{E}_{t,1}) + \beta Int(\boldsymbol{E}_{t,2}))||_1. \tag{6}$$

This penalty can be viewed as promoting faithful reconstruction of the pseudobond length between C_α atoms, as well as a smooth interpolation of these lengths along paths in L_I. This penalty is analogous to the metric preservation regularizer introduced in (Cosmo et al., 2020) for 3D meshes. Importantly, this promotes the generation of a protein whose metric of the backbone trace is invariant to the extrinsic latent code, $\boldsymbol{z}_e$. Thus, the loss function $\mathcal{L}$ for ProGAE is,

$$\mathcal{L}((\hat{\boldsymbol{P}}_1, \hat{\boldsymbol{P}}_2), (\boldsymbol{P}_1, \boldsymbol{P}_2)) := \sum_{i=1}^{2} SmoothL_1(\hat{\boldsymbol{P}}_i, \boldsymbol{P}_i) + \lambda_R \mathbb{E}_{\beta \in \mathbf{U}[0,1]} \mathcal{R}(\hat{\boldsymbol{P}}_1, \hat{\boldsymbol{P}}_2; \beta). \tag{7}$$



(a) 3 RBDs of S protein

(b) Ectodomain of Human ACE2

Figure 2: Reconstructions of protein frames from test data using ProGAE. The top row displays the ground truth (each frame belongs to a different trajectory), while the bottom row displays corresponding structure generated by the network. Color in the top row denotes separate protein chains, while color in the bottom row indicates the log of atom-wise L_2 error. Color of the bonds indicates the average of the constituent atoms.

Table 1: Average atom-wise L_2 error and Bond length error on training and test set using ProGAE. The L_2 error is within the resolution of the associated PDB files, even achieving sub-Angstrom performance on human ACE2.

		S protein	**hACE2**
Resolution (Å)		2.68/2.80	2.68
Training	Reconstruction error	8.28 ± 0.04 E-1	4.29 ± 0.01 E-1
	Bond length error (Å)	3.71 ± 0.02 E-1	1.56 ± 0.01 E-1
	Avg. L_2 error (Å)	1.21 ± 0.00	0.78 ± 0.01
Test	Reconstruction error	1.39 ± 0.01 E0	6.27 ± 0.03 E-1
	Bond length error (Å)	3.88 ± 0.02 E-1	1.63 ± 0.02 E-1
	Avg. L_2 error (Å)	1.56 ± 0.01	0.90 ± 0.01

3 Experimental Setup

In this section, we describe the setup of our numerical experiments confirming the usefulness of ProGAE in generating protein dynamics. For each dataset, we train three models, each from a different random seed, and report both mean and standard deviation in our results. Details on the choice of network hyperparameters can be found in the Appendix A.2.

Datasets Datasets used in this work are atomistic simulation trajectories [1] (D.E. Shaw Research, 2020). The two main datasets we use are simulations of proteins in presence of FDA approved or under-investigation molecules. These are: (1) 50 independent trajectories, each simulating the SARS-CoV-2 trimeric spike protein (S protein) in presence of a distinct drug for 2μs. The simulation is limited to 3 receptor binding domains (RBDs) of the S protein, as well as another short region needed for the system to maintain a trimer assembly; (2) 75 independent trajectories, each simulating the ectodomain protein of human ACE2 (hACE2) in presence of a distinct drug for 2μs.

The S protein and the hACE2 protein contains 3,690 backbone atoms and 2,386 backbone atoms, respectively. The time resolution is 1,200 ps. We use the first 70% of frames from each trajectory to form the training set. The next 10% and the last 20% of frames form the validation set and test set, respectively. This split tests the ability of the trained model to *extrapolate or recovery* from the dynamics present in the first part of the trajectory to the last part of the trajectory.

For transfer learning experiments, we also consider two additional trajectories of the entire S protein containing 13,455 backbone atoms. One trajectory is initiated from a closed state, while the other

[1] available here: `http://www.deshawresearch.com/resources_sarscov2.html`

Table 2: The leading canonical correlation between the learned intrinsic and extrinsic latent spaces. And, the performance of a linear model trained on disentangled latent spaces of ProGAE for bound drug classification is shown.

	S protein	**hACE2**
First canonical correlation	0.08 ± 0.00	0.07 ± 0.01
Drug classifier on intrinsic latent space	$2.04 \pm 0.04\%$	1.43 ± 0.01 %
Drug classifier on extrinsic latent space	$99.65 \pm 0.26\%$	99.62 ± 0.21 %

Table 3: Results of linear regression models trained on the extrinsic latent space for predicting certain physical and chemical properties of the drugs that a protein is bound to. Error is normalized for interpretability. For comparison, performance of a linear regression model trained on the PCA embeddings of the orientation of the backbone bonds is reported. This embedding is restrained to have the same dimension as the latent space.

Dataset		Molecular weight	Hydrogen bond donor count	Topological polar surface area
S protein	PCA error (σ)	0.78 ± 0.00	0.81 ± 0.01	0.79 ± 0.00
	Latent error (σ)	$\mathbf{0.55 \pm 0.04}$	$\mathbf{0.56 \pm 0.03}$	$\mathbf{0.61 \pm 0.00}$
hACE2	PCA error (σ)	0.71 ± 0.00	0.65 ± 0.00	0.73 ± 0.00
	Latent error (σ)	$\mathbf{0.55 \pm 0.01}$	$\mathbf{0.57 \pm 0.01}$	$\mathbf{0.53 \pm 0.02}$

is initiated from a partially open state. We use the first 2.5 μs of these 10 μs simulations for experiments, corresponding to 2,001 frames with a time resolution of 1,200 ps. Additionally, we utilize the first 10 μs of a 100 μs simulation of the main Protease of SARS-CoV-2. This corresponds to a sequence of 10,001 frames with a 1,000 ps time resolution.

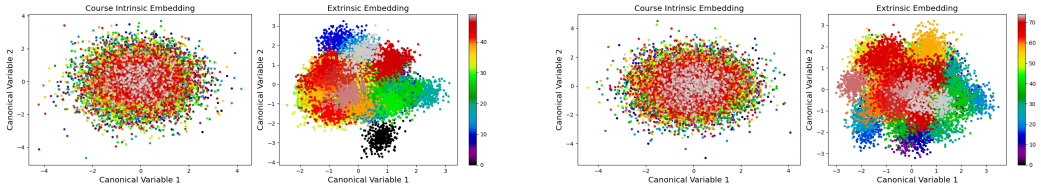
4 RESULTS

Structure Reconstruction (or Recovery) Figures 2a and 2b display the ability of ProGAE to accurately reconstruct protein trajectories for both the S protein and hACE2. From the log of atom-wise L_2 error mapped on bonds, as well as from visual comparison, it is clear that our network can capture and reconstruct notable conformational changes of a protein. Figures 5a and 5b in the appendix display these reconstructions with color denoting fragment instead of L_2 error. Consistent with low L_2 error, visually the reconstructed backbones appear consistent with ground truths, with larger deviations observed in the flexible loop regions.

Table 1 contains performance metrics of ProGAE on training and test sets. Generalization is measured by the L_2 reconstruction error of the coordinates of the backbone atoms. For hACE2, we are able to achieve sub-Angstrom performance on the test set. In either case, the reconstruction is within experimental resolution of the associated PDB files; 6VXX/6VW1 for the S protein and 6VW1 for hACE2. Additionally, the average error in the length of the pseudobonds is also sub-Angstrom. For reference, the average pseudobond in the datasets has a length of 3.86Å. Thus, it is evident that ProGAE is able to reconstruct proteins within meaningful resolution.

Utility of the Extrinsic Latent Space With the reconstruction capabilities of ProGAE verified, we consider the benefit of having separate intrinsic and extrinsic latent spaces. First, we explore the statistical relationship between the learned intrinsic latent space and the extrinsic latent space. Canonical correlation analysis (CCA) is a natural approach to study if a linear relationship exists (Hardoon et al., 2004). We include background on CCA in Appendix A.1.

Table 2 includes the leading correlation between the intrinsic and extrinsic latent spaces for each dataset. Note this correlation is very low, implying that there is a negligible linear relationship between the intrinsic and extrinsic latent spaces. Learning a disentangled latent representation is often desired for better interpretability, which is typically measured using a generalization of mutual information such as total correlation as in (Chen et al., 2018). While structural conditions may



(a) S protein

(b) Human ACE2

Figure 3: The embedding of the data in the projection of the latent space to the first two canonical vectors between the intrinsic and extrinsic latent space. Color indicates the identity of the drug that the protein is bound to within that frame. Clustering by drug identity is apparent in the extrinsic latent space, but not the intrinsic latent space.

prevent the intrinsic and extrinsic embeddings from being completely independent, Table 2 indicates absence of any strong linear relationship between intrinsic and extrinsic latent vectors in our learned model, confirming disentanglement.

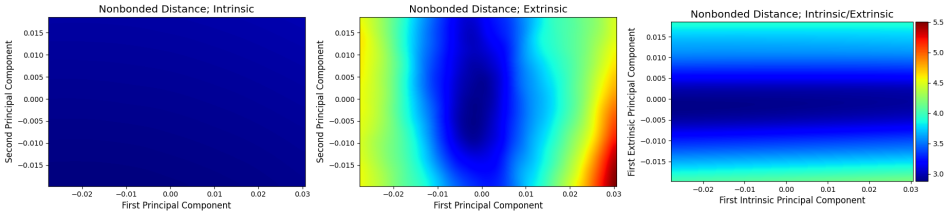
As stated earlier, each trajectory in the dataset corresponds to the S or hACE2 protein bound to a specific drug. Then it is natural to investigate if this distinct drug information is encoded in the two disentangled latent spaces. Table 2 contains the performance of a linear classifier trained on the different latent spaces to classify the drug present in each frame. It is clear that the drug molecule can be almost perfectly classified in the extrinsic latent space, while such classification is random in the intrinsic latent space. Figures 3a and 3b display the embeddings of the test set in the latent spaces, projected to the first two canonical components. In these figures, color denotes the identity of the drug that the protein is bound to. Even in the 2D projection of the extrinsic latent space, clustering by the drug identity is apparent, which is not the case for the intrinsic embedding.

Next, we consider if this linear separation is chemically meaningful. To this end, we train a linear regression model on the extrinsic latent space to predict certain physico-chemical properties of a drug binding to a protein. Table 3 displays the performance of the regression model at predicting the properties of molecular weight, hydrogen bond donor count, and topological polar surface area. For comparison, we train a linear regression model on the first n_e principal component scores corresponding to the PCA of the extrinsic signal on each element of the test dataset. The latent regression outperforms that of PCA, indicating that the the extrinsic latent embedding captures more physico-chemical information about the bound drug.

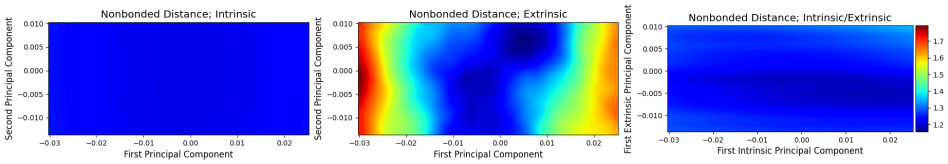
Geometry of the Generated Dynamics As a generative model, it is important to consider if the dynamics generated via paths in the latent space match our expectations. One way in which we measure this is how the distance matrix of atoms in a generated protein changes as the latent variables change. Specifically, we are interested in the distance between non-bonded atoms.

In Figures 4a and 4b, we plot the norm of the difference between these distance matrices for the generated protein and that of the protein at its initiation. We sample the latent space along two principal components, while setting others to 0. So in these figures 4a and 4b, we see three different cross sections of this space. It is immediately apparent that a change in the extrinsic latent code dominates changes in the non-bonded distance. This is a desired behavior of a disentangled geometric autoencoder, where large-scale changes are captured in the extrinsic latent space, while small-scale changes in local distance are controlled by the intrinsic latent space.

Transfer Learning– Extension to Different Proteins To check the generalization power of ProGAE, we consider transfer learning to trajectories of completely different proteins. We begin with models trained on the S protein comprised of the 3 RBDs and on hACE2. These results are summarized in Table 4. We consider the transfer of learned ProGAE models to trajectories of the closed and partially open state of the entire S protein, as well as SARS-COV-2 main Protease, which provides insight into the generalization capability of the convolutional filters that we have learned. As a result, six scenarios of the transfer learning, in addition three random baselines, are reported in Table 4. When transferring the model trained on the 3 RBDs of the S protein to the S protein in the closed state, we are transferring the model learned on a partial structure to the entire protein



(a) S protein



(b) Human ACE2

Figure 4: Error of the non-bonded distance between the backbone atoms for generated structures compared to the protein at its initiation. Each point in the plot corresponds to a protein structure generated by sampling the latent space via the plane of the two specified principal components of the latent spaces. The choice of the intrinsic latent code hardly affects the non-bonded distance error, unlike the extrinsic latent code.

Table 4: Average atom-wise L_2 error in reconstruction (Å) on the test dataset after transferring a trained model to other protein trajectories. These trajectories are two different states of the entire S protein and of the main protease of SARS-CoV-2. We retrain only the dense layer mapping to the decoder. For comparison, we train the same layer for a randomly initialized model (Baseline).

Source dataset	**Closed S**	**Partially Open S**	**Protease**
Baseline (Å)	1.79 ± 0.04	2.26 ± 0.03	1.36 ± 0.03
S protein (Å)	1.55 ± 0.01	2.01 ± 0.07	$\mathbf{1.16 \pm 0.00}$
hACE2 (Å)	$\mathbf{1.52 \pm 0.03}$	$\mathbf{1.98 \pm 0.00}$	1.17 ± 0.01

that is much larger in size. Model transfer to the S protein in the partially open state deals with a scenario where the conformational state of the protein is notably different (closed vs partially open). Transferring the model trained on hACE2 to the S protein datasets studies the knowledge transfer to an entirely different protein, but one which hACE2 is known to interact with. Finally, transferring both S protein and hACE2 models to the main Protease trajectory allows us to study the transfer of the models to a completely different protein without notable interaction with the source protein.

The only incompatible layer is the dense layer mapping from the latent spaces. To investigate transfer learning, we train just this dense layer for 10 epochs. As a baseline, we train the same layer of a randomly initialized model. In all cases, the transferred model performs better than the baseline. Thus the learned filters generalize to trajectories of completely different protein systems.

5 Conclusion

In this work we introduce a novel geometric autoencoder named ProGAE for learning meaningful disentangled representations of the protein structural dynamics. Our model accurately reconstructs the protein dynamics. The autoencoder framework separately encodes intrinsic and extrinsic geometries to ensure better latent space interpretability. The extrinsic latent space can classify the protein structures with respect to the bound drug molecules, as well as can predict the drug properties. We also show that the geometric convolutional filters learned in training can be successfully transferred to trajectories of different protein systems, irrespective of system size, conormational state, or presence of protein-protein interaction. These results on learning, predicting, and generating protein dynamics suggest that the proposed framework can serve as the first step towards bridging geometric deep learning with molecular dynamics.

REFERENCES

Debsindhu Bhowmik, Shang Gao, Michael T Young, and Arvind Ramanathan. Deep clustering of protein folding simulations. *BMC bioinformatics*, 19(18):47–58, 2018.

Luigi Bonati, Yue-Yu Zhang, and Michele Parrinello. Neural networks-based variationally enhanced sampling. *Proceedings of the National Academy of Sciences*, 116(36):17641–17647, 2019.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pp. 2610–2620, 2018.

Wei Chen and Andrew L Ferguson. Molecular enhanced sampling with autoencoders: On-the-fly collective variable discovery and accelerated free energy landscape exploration. *Journal of computational chemistry*, 39(25):2079–2102, 2018.

Wei Chen, Hythem Sidky, and Andrew L Ferguson. Nonlinear discovery of slow molecular modes using state-free reversible vampnets. *The Journal of chemical physics*, 150(21):214114, 2019.

Zhengdao Chen, Jianyu Zhang, Martin Arjovsky, and Léon Bottou. Symplectic recurrent neural networks, 2020.

Luca Cosmo, Antonio Norelli, Oshri Halimi, Ron Kimmel, and Emanuele Rodolà. Limp: Learning latent shape representations with metric preservation priors. *arXiv preprint arXiv:2003.12283*, 2020.

D.E. Shaw Research. Molecular dynamics simulations related to sars-cov-2. `http://www.deshawresearch.com/resources_sarscov2.html`, 2020. Accessed: 2020-09-30.

Matteo T Degiacomi. Coupling molecular dynamics and deep learning to mine protein conformational space. *Structure*, 27(6):1034–1040, 2019.

Manfredo P Do Carmo. *Differential geometry of curves and surfaces: revised and updated second edition*. Courier Dover Publications, 2016.

P Gainza, F Sverrisson, F Monti, E Rodola, MM Bronstein, and BE Correia. Deciphering interaction fingerprints from protein molecular surfaces. *bioRxiv*, pp. 606202, 2019.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

Liyu Gong and Qiang Cheng. Exploiting edge features for graph neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9211–9219, 2019.

Jordan Graves, Jacob Byerly, Eduardo Priego, Naren Makkapati, S Vince Parish, Brenda Medellin, and Monica Berrondo. A review of deep learning methods for antibodies. *Antibodies*, 9(2):12, 2020.

Xiaojie Guo, Sivani Tadepalli, Liang Zhao, and Amarda Shehu. Generating tertiary protein structures via an interpretative variational autoencoder. *arXiv preprint arXiv:2004.07119*, 2020.

David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664, 2004.

Pedro Hermosilla, Marco Schäfer, Matěj Lang, Gloria Fackelmann, Pere Pau Vázquez, Barbora Kozlíková, Michael Krone, Tobias Ritschel, and Timo Ropinski. Proteinn: Intrinsic-extrinsic convolution and pooling for scalable deep protein analysis, 2020.

John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. In *Advances in Neural Information Processing Systems*, pp. 15820–15831, 2019a.

John Ingraham, Adam J Riesselman, Chris Sander, and Debora S Marks. Learning protein structure with a differentiable simulator. In *ICLR*, 2019b.

Mariusz Jaskolski, Miroslaw Gilski, Zbigniew Dauter, and Alexander Wlodawer. Stereochemical restraints revisited: how accurate are refinement targets and how much should protein structures be allowed to deviate from them? *Acta Crystallographica Section D: Biological Crystallography*, 63(5):611–620, 2007.

Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael J. L. Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020.

Hyungro Lee, Heng Ma, Matteo Turilli, Debsindhu Bhowmik, Shantenu Jha, and Arvind Ramanathan. Deepdrivemd: Deep-learning driven adaptive molecular simulations for protein folding, 2019.

Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. Vampnets for deep learning of molecular kinetics. *Nature Communications*, 9(1), Jan 2018. ISSN 2041-1723. doi: 10.1038/s41467-017-02388-1. URL `http://dx.doi.org/10.1038/s41467-017-02388-1`.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

Frank Noé, Gianni De Fabritiis, and Cecilia Clementi. Machine learning for protein folding and dynamics. *Current Opinion in Structural Biology*, 60:77–84, 2020a.

Frank Noé, Alexandre Tkatchenko, Klaus-Robert Müller, and Cecilia Clementi. Machine learning for molecular simulation. *Annual review of physical chemistry*, 71:361–390, 2020b.

Venkata K. Ramaswamy, Chris G. Willcocks, and Matteo T. Degiacomi. Learning protein conformational space by enforcing physics with convolutions and latent interpolations, 2020.

João Marcelo Lamim Ribeiro, Pablo Bravo, Yihang Wang, and Pratyush Tiwary. Reweighted autoencoded variational bayes for enhanced sampling (rave). *The Journal of chemical physics*, 149 (7):072301, 2018.

N. Joseph Tatro, Stefan C. Schonsheck, and Rongjie Lai. Unsupervised geometric disentanglement for surfaces via cfan-vae, 2020.

Sun-Ting Tsai, En-Jui Kuo, and Pratyush Tiwary. Learning molecular dynamics with simple language model built upon long short-term memory neural network, 2020.

Yasemin Bozkurt Varolgüneş, Tristan Bereau, and Joseph F Rudzinski. Interpretable embeddings from molecular simulations using gaussian mixture variational autoencoders. *Machine Learning: Science and Technology*, 1(1):015012, 2020.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Wujie Wang, Simon Axelrod, and Rafael Gómez-Bombarelli. Differentiable molecular simulations for control and learning, 2020.

Wayne Wu, Kaidi Cao, Cheng Li, Chen Qian, and Chen Change Loy. Disentangling content and style via unsupervised geometry distillation, 2019.

Jie Yang, Kaichun Mo, Yu-Kun Lai, Leonidas J. Guibas, and Lin Gao. Dsm-net: Disentangled structured mesh net for controllable generation of fine geometry, 2020.

Jun Zhang, Yi Isaac Yang, and Frank Noé. Targeted adversarial learning optimized sampling. *The journal of physical chemistry letters*, 10(19):5791–5797, 2019.