Boltzmann generators: Efficient sampling of equilibrium states of many-body system with flow-based generative models

Wei-Tse Hsu[1,*] and Theodore Fobe[1,*]

[1]*Department of Chemical and Biological Engineering, University of Colorado Boulder, Boulder, CO 80305*

Owing to the incredibly small volume occupied by the equilibrium states in the configurational space, generating statistically independent samples of equilibrium states of many-body systems has been regarded as a significant challenge in the field of computational molecular science. The nature of equilibrium states necessitates the utilization of small steps in molecular simulations (e.g. molecular dynamics (MD) simulations or Monte Carlo (MC) simulations), making it nearly impossible to comprehensively sample the region of interest within a reasonable amount of time. Combining deep learning and statistical mechanics, Boltzmann generators circumvent this sampling issue by repacking the high probability regions (i.e. equilibrium states) of configuration space into a concentrated region in latent space. In this study, we employ Boltzmann generators to different systems in an attempt to generate statistically independent samples. As a consequence, we show that Boltzmann generators are able to generate Boltzmann-weighted samples and accurately compute free energy profiles along reaction coordinates in a variety of systems. The simple linear interpolation in latent space also provides additional molecular insights into the system. In addition, comparisons between Boltzmann generators trained on different loss functions are demonstrated.

I. INTRODUCTION

In the past decades, molecular simulations, including molecular dynamics (MD) simulations and Monte Carlo (MC) simulations, have played a crucial role in various disciplines of science, including biophysics,[1] pharmaceutical chemistry[2] and material science.[3] However, the usefulness of classical molecular simulations is severely restricted by kinetic bottlenecks as a result of systems characterized by a rough free energy surface. For most systems, metastable states separated by numerous energy barriers typically occupy vanishingly small volume in phase space, leading to prohibitive computational cost for generating statistically independent samples from the Boltzmann distribution.

To address this challenge of phase space sampling, over the years, an enormous amount of research has been devoted to the development of advanced sampling methods. While these methods generally can mitigate the problem, they also have their drawbacks. For example, for sampling techniques such as metadynamics,[4] umbrella sampling,[5] adaptive biasing force[6] or their variants,[7,8] it is required to define reasonable reaction coordinates (RC) that can capture all the slow degrees of freedom of the system, which could be particularly challenging for complicated condensed matter systems. Although methods such as replica exchange molecular dynamics (REMD),[9,10] expanded ensemble[11] and their variants[12,13]do not require predefined reaction coordinates, they do require a series of alchemical intermediate states to bridge the gap in probability overlap between different metastable states. Overall, all these methods fail to draw statistically independent samples from Boltzmann-type distributions in one shot to compute statistical observables of the systems, such as free energy differences.

Boltzmann generators,[14] in contrast, do not require any knowledge of reaction coordinates, nor the intermediate thermodynamic states, successfully integrating the strengths of deep learning and statistical mechanics. In a Boltzmann generator, we train an invertible neural network to learn the transformation of coordinates from configuration space $\mathbf{x}$ to the so-called latent space $\mathbf{z}$. To ensure the adjacency between the low-energy configurations from different equilibrium states in latent space, we adopt a Gaussian distribution $\mu_z(\mathbf{z})$ as the prior distribution such that high-probability configurations are mapped to the center of latent space and therefore can be easily sampled.

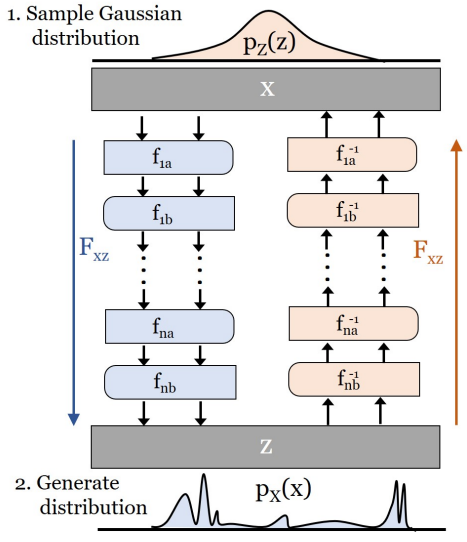

FIG. 1. The architecture of a Boltzmann generator. The figure was adapted from the work[14] by Noé *et al.*

* These two authors contributed equally to this work

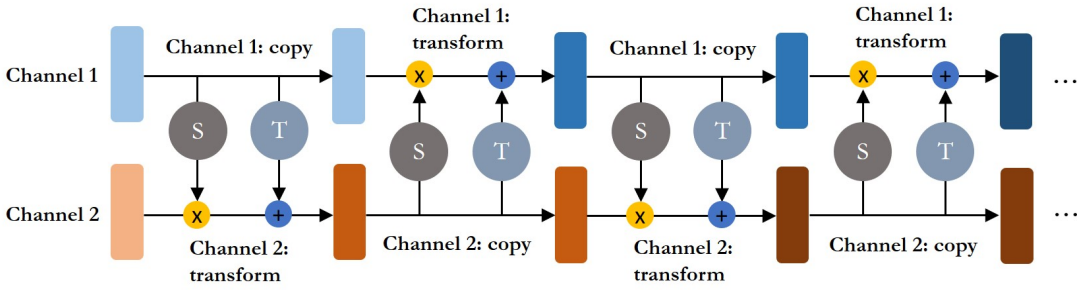

FIG. 2. The structure of affine coupling layers in RealNVP blocks.

As shown in Figure 1, in a Boltzmann generator, the configuration variable $\mathbf{x}$ is transformed into the latent variable $\mathbf{z}$ by a deep neural network F_{xz} (the so-called inverse generator) composed of a series of invertible transformation blocks f_1, ..., f_n, which are known as real-valued non-volume-preserving (RealNVP) blocks. Conversely, to generate samples in configuration space that approximates the Boltzmann distribution, the network F_{zx} (generator) maps the samples $\mathbf{z}$ drawn from the Gaussian prior distribution $\mu_z(\mathbf{z})$ back to the configuration space.

In this study, we demonstrate the usage of Boltzmann generators in generating statistically independent samples in different many-body systems, including 2-dimensional systems such as the double-well potential and the Müller-Brown (MB) potential, and a higher-dimensional system, which is a bistable dimer immersed in a Lennard-Jones fluid. Incorporated with the consideration of certain reaction coordinates, these generated samples enable accurate computations of the free energy profile. In addition, with a trained Boltzmann generator, we show that realistic transition pathways can also be predicted by simple linear interpolations in latent space.

II. THEORY

A. Real-valued non-volume-preserving transformation

As an application of flow-based generative models, a Boltzmann generator implements real-valued non-volume-preserving (RealNVP) transformations[15] in a deep neural network and its inverse, which transform the probability densities incrementally from configuration space to latent space or from latent space to configuration space. Given the trainable parameters θ, Boltzmann-distributed random variables $\mathbf{x} = F_{zx}(\mathbf{z};\theta)$ and Gaussian random variables $\mathbf{z} = F_{xz}(\mathbf{x};\theta)$, the Jacobian matrices of the transformation F_{zx} and its inverse (F_{xz}) can be expressed as:

$$\mathbf{J}_{zx}(\mathbf{z};\theta) = \left[\frac{\partial F_{zx}(\mathbf{z};\theta)}{\partial z_1}, \ ..., \ \frac{\partial F_{zx}(\mathbf{z};\theta)}{\partial z_n}\right] \tag{1}$$

$$\mathbf{J}_{xz}(\mathbf{x};\theta) = \left[\frac{\partial F_{xz}(\mathbf{x};\theta)}{\partial x_1}, \ ..., \ \frac{\partial F_{xz}(\mathbf{x};\theta)}{\partial x_n}\right] \tag{2}$$

Since the absolute value of the Jacobian's determinant (e.g. $|\det \mathbf{J}_{zx}(\mathbf{z};\theta)|$) measures how much a volume element at the original space is scaled by the transformation, by the change of variables theorem and the inverse function theorem, we can write:

$$\begin{aligned} p_X(\mathbf{x}) &= p_z\left(F_{xz}(\mathbf{x})\right) R_{xz}(\mathbf{x}) \\ &\Rightarrow \log p_X(\mathbf{x}) = \log p_z\left(F_{xz}(\mathbf{x})\right) + \log R_{xz}(\mathbf{x}) \end{aligned} \tag{3}$$

$$\begin{aligned} p_Z(\mathbf{z}) &= p_x\left(F_{zx}(\mathbf{z})\right) R_{zx}(\mathbf{z}) \\ &\Rightarrow \log p_Z(\mathbf{z}) = \log p_x\left(F_{zx}(\mathbf{z})\right) + \log R_{zx}(\mathbf{z}) \end{aligned} \tag{4}$$

To ensure a low computational cost for training a Boltzmann generator, the computation of the transformations (F_{xz} and F_{zx}) and their determinants of the Jacobian matrices (R_{xz} and R_{zx}) shown in Equations 3. and 4., must be efficient. In a RealNVP model, this is achieved by the two affine coupling layers (f_{ia} and f_{ib} for the block f_i in Figure 1). that comprise one RealNVP block. Specifically, in each affine coupling layer, the input dimensions are split into two channels ($\mathbf{x}_1 = \mathbf{x}_{1:d}$ and $\mathbf{x}_2 = \mathbf{x}_{d+1:D}$), where the first d dimensions remain the same in the first channel, and the remaining dimensions (from $d+1$ to D) undergo an affine transformation (scale-and-shift transformation) in the second channel accomplished by the scaling (S) and translating (T) functions:

$$\begin{aligned} &f_{xz}(\mathbf{x}_1, \mathbf{x}_2): \\ &\quad \begin{cases} \mathbf{z}_1 = \mathbf{x}_1 \\ \mathbf{z}_2 = \mathbf{x}_2 \odot \exp(S(\mathbf{x}_1;\theta)) + T(\mathbf{x}_1;\theta) \end{cases} \end{aligned} \tag{5}$$

$$\begin{aligned} &f_{zx}(\mathbf{z}_1, \mathbf{z}_2): \\ &\quad \begin{cases} \mathbf{x}_1 = \mathbf{z}_1 \\ \mathbf{x}_2 = (\mathbf{z}_2 - T(\mathbf{x}_1;\theta)) \odot \exp(-S(\mathbf{z}_1;\theta)) \end{cases} \end{aligned} \tag{6}$$

As shown in Equation 5. and 6., this design of affine coupling layers has two advantages:

- The transformations (f_{xz} and f_{zx}) and their corresponding inverse (f_{zx} and $_{xz}$) are both straightforward. In addition, the computation of R_{xz} and R_{zx} are computationally cheap. For example,

$$\begin{aligned} \mathbf{J}_{xz} &= \begin{bmatrix} \mathbf{I}_d & \mathbf{0}_{d\times(D-d)} \\ \frac{\partial \mathbf{z}_{d+1:D}}{\partial \mathbf{x}_{1:d}} & \text{diag}(\exp(S(\mathbf{x}_{1:d}))) \end{bmatrix} \\ \Rightarrow \log R_{xz} &= \sum_{j=1}^{D-d} S(\mathbf{x}_{1:d})_j \end{aligned} \tag{7}$$

- Since f_{xz} and f_{zx} do not require computing the inverse of S or T and computing R_{zx} and R_{xz} does not involve computing the Jacobian matrices of S or T, the functions S and T can as complicated as needed; i.e. both S and T can be modeled by deep neural networks.

Figure 2. illustrates the sequential transformation of configuration space samples to the latent space described above. In each RealNVP block the model alternates between the duplication and the affine transformation of both channels of data to ensure forward and reverse transformations occur in a reversible manner.

B. Sampling

Samples of the Boltzmann distribution are generated using Metropolis Monte Carlo (MC) simulations. As described in the original paper, the use of a small local step size is meant to emulate molecular dynamic simulations, ensuring that individual simulations stay trapped in the metastable states of interest. This method of generating "MD" simulation samples generates configuration space samples $\mathbf{x}$, which can be directly used to train Boltzmann generators.

In each MC step, the system of interest is perturbed with an isotropic normal distribution scaled by a system-dependent σ_{Metro}. The proposed configuration is accepted or rejected using the standard Metropolis criterion, shown in Equation 8.

$$\alpha(x^*|x_{i-1}) = \min\left[1, \exp\left[\frac{-(u(\mathbf{x}^*) - u(\mathbf{x_{i-1}}))}{k_B T}\right]\right] \tag{8}$$

Where, $u(\mathbf{x})$ represents the energy of a given configuration and $k_B T$ represents the reduced temperature of the system.

C. Training a Boltzmann generator

As Noé *et al.* suggest in the original paper,[14] Boltzmann generators are trained by combining two modes: training by examples and training by energy, where the former utilizes samples drawn from the configuration space by Monte Carlo or molecular dynamic simulations to train the neural networks and the latter uses the samples drawn from the prior Gaussian distribution in latent space.

Specifically, during the training, the parameters of the neural networks are adjusted such that the distance between the exact distribution (μ) and the generated distribution (q) can be minimized, in either the latent or configuration space. This distance can be measured by the Kullback-Leibler (KL) divergence. For example, KL divergence in latent space measuring the distance from the prior Gaussian distribution $\mu_Z(\mathbf{z})$ to the generated distribution in latent space $q_Z(\mathbf{z})$ can be written as:

$$KL_\theta(\mu_Z \| q_Z) = \int \mu_Z(\mathbf{z}) \log\left(\frac{\mu_Z z(\mathbf{z})}{q_Z(\mathbf{z})}\right) d\mathbf{z} \tag{9}$$

Given that

$$\mu_Z(\mathbf{z}) = \frac{\mathrm{e}^{-u_z(\mathbf{z})}}{Z_Z} = \frac{\mathrm{e}^{-\frac{1}{2}(\frac{\mathbf{z}}{\sigma})^2}}{\sigma\sqrt{2\pi}} \tag{10}$$

and

$$\mu_X(\mathbf{x}) = \frac{\mathrm{e}^{-u_x(\mathbf{x})}}{Z_X} \tag{11}$$

Equation 9. can be derived to

$$\begin{aligned} &KL_\theta(\mu_Z \| q_Z) \\ &= -H_z + \log Z_X + \mathbb{E}_{\mathbf{z}\sim\mu_Z(\mathbf{z})}\left[u_X(F_{xz}(\mathbf{z}) - \log R_{zx}(\mathbf{z})\right] \end{aligned} \tag{12}$$

Because H_Z and Z_X are constant in θ, we define the KL loss J_{KL} as follows:

$$J_{KL} = \mathbb{E}_{\mathbf{z}\sim\mu_Z(\mathbf{z})}\left[u_X(F_{zx}(\mathbf{z}) - \log R_{zx}(\mathbf{z})\right] \tag{13}$$

Accordingly, when the Boltzmann generator is trained by energy, it adjusts the parameters in F_{zx} (hence the parameters of the inverse of F_{xz}, F_{zx}) to minimize the KL loss J_{KL}. Interestingly, the first term of the KL loss, which is the internal energy of the system, counteracts the second term (an effective entropic contribution to the free energy) such that the model tries to sample low-energy configurations to minimize $u(F_{zx}(\mathbf{z}))$ and simultaneously penalize the system for collapsing into a single metastable state to maximize the entropy of the generated distribution $\log R_{zx}(\mathbf{z})$.

However, it was validated[14] that the entropy term in the KL loss is not sufficient to prevent the so-called mode collapse, which necessitates the introduction of the maximum likelihood (ML) loss function (J_{ML}), i.e. training by examples. Mathematically, J_{ML} can be derived by either maximizing the probability of configuration samples $\mathbf{x}$ in the Gaussian distribution or minimizing the distance between the generated distribution in configuration space $q_X(\mathbf{x})$ and the Boltzmann distribution $\mu_X(\mathbf{x})$, i.e. the KL divergence in the configuration space. J_{ML} can be expressed as:

$$J_{ML} = \mathbb{E}_{\mathbf{x}\sim\rho(\mathbf{x})}\left[\frac{1}{2\sigma^2}\|F_{xz}(\mathbf{x})\|^2 - \log R_{xz}(\mathbf{x})\right] \tag{14}$$

where σ is the standard deviation of the prior Gaussian distribution. Note that since we are not able to sample from the Boltzmann distribution $\mu(\mathbf{x})$ *a priori*, we instead approximate it by a sampling $\rho(\mathbf{x})$, the distribution represented by the MC samples used to train the Boltzmann generator.

While training a Boltzmann generator on the KL loss and ML loss enables sampling of high-probability states, sometimes we want to sample low-probability (high-energy) states such as a transition state along a predefined reaction coordinate. In this situation, we have to include the reaction coordinate (RC) loss, which can be defined as:

$$\begin{aligned} J_{RC} &= \int p\left(r(\mathbf{x})\right) \log p\left(r(\mathbf{x})\right) dr(\mathbf{x}) \\ &= \mathbb{E}_{\mathbf{x}\sim q_X(\mathbf{x})} \log p\left(r(\mathbf{x})\right) \end{aligned} \tag{15}$$

where $r(\mathbf{x})$ is the reaction coordinate. Practically, to accurately compute $p\left(r(\mathbf{x})\right)$, we employ batchwise kernel density estimation (KDE)[16,17] with K-fold cross validation optimizing the bandwidth of the Gaussian kernel functions.

Finally, considering all kinds of loss functions, we define the total loss function J_{tot}:

$$J_{tot} = w_{ML}J_{ML} + w_{KL}J_{KL} + w_{RC}J_{RC} \tag{16}$$

where the w's are the weights of the loss functions. In practice, typically it takes dozens of epochs/hundreds of iterations to converge the loss function.

D. Latent interpolation

As Noé *et al.* proposed in the original paper,[14] a trained Boltzmann generator can provide some insights into the reaction pathways between metastable states by mapping the paths in latent space back to configuration space. Specifically, in each system of interest, we first randomly choose 10 configurations near the local minimum. Then, we pair up these 20 configuration samples and transform them to latent space, linearly interpolating between the latent space representations of samples from different energy minima. In the end, we map the 10 interpolated paths in latent space back to configuration space, resulting in non-linear reaction paths in configuration space between the energy minima.

E. Free energy calculations

In view of the fact that the generated samples might be more or less biased by the neural network, to obtain unbiased samples to compute Boltzmann-weighted averages (like the free energy of the system), we have to reweight the generated distribution $q_x(\mathbf{x})$ to the Boltzmann distribution. Therefore, we define a statistical weight $w_x(\mathbf{x})$ such that:

$$w_x(\mathbf{x}) = \frac{\mu_x(\mathbf{x})}{q_x(\mathbf{x})} = \frac{q_z(\mathbf{z})}{\mu_z(\mathbf{z})} \tag{17}$$

and

$$w_x(\mathbf{x}) \propto e^{-u_x(F_{xz}(\mathbf{z}))+u_z(\mathbf{z})+\log R_{zx}(\mathbf{z};\theta)} \tag{18}$$

Then, we apply this statistical weight in the kernel density estimation mentioned in the last section or in a weight histogram such that the probability of i-th bin p_i can be estimated by:

$$p_i = \frac{\sum_{k=1}^{n_i} w_{i,k} \cdot x_{i,k}}{\sum_{i=1}^{m} n_i} \tag{19}$$

where m and n_i are the number of bins and the number of events in the i-th bin, respectively, $x_{i,k}$ is the k-the sample in the i-the bin, and $w_{i,k}$ is its corresponding statistical weight. Using $p\left(r(\mathbf{x})\right)$ calculated by either weighted histograms or KDE, we can calculate the reduced free energy as follows:

$$f\left(r(\mathbf{x})\right) = -\log p\left(r(\mathbf{x})\right) \tag{20}$$

Since we don't know the absolute free energy value, when plotting a free energy profile, we subtract $f\left(r(\mathbf{x})\right)$ by its minimum to take $f = 0$ as the reference.

III. APPLICATIONS

A. System 1: Double-well Potential (Wei-Tse)

In this study, instead of using Keras[18] and TensorFlow[19] as the original paper,[14] we implemented Boltzmann generators in PyTorch.[20] Our implementation and relevant examples are hosted in a GitHub repo. To ensure the efficacy of our implementation, we start with the double-well potential model, which can be expressed as:

$$u(\mathbf{x}) = u(x_1, x_2) = ax_1^4 - bx_1^2 + cx_1 + dx_2^2 \tag{21}$$

As shown in Figure 3., we adopted $(a,\ b,\ c,\ d) = (1,\ 6,\ 1,\ 1)$ and the two-dimensional potential model exhibits two metastable states separated by a high energy barrier. x_2 is used as the reaction coordinate since it is the slow degree of freedom of the system. To acquire the dataset for training the Boltzmann generator, we performed a 5000-step Monte Carlo simulation for each well starting from its local minimum with the reduced temperature set as 1. To preserve the Boltzmann weights in the samples, we include the configuration at every Monte Carlo step no matter the MC move is accepted or not. As a result, there are 10002 configuration samples in the training dataset.

For this simple test system, we build a Boltzmann generator composed of 3 RealNVP blocks where there

are two affine coupling layers in each block. In an affine coupling layer, there is a neural network composed of three layers for each of the transformation functions (S and T). In the neural network, there are 100 nodes in each layer and the activation functions include the ReLU function and the hyperbolic tangent function. In addition, we used Adam optimizer in the gradient descent method with the learning rate being 0.001. In the end, to train the Boltzmann generator, we include 2048 samples for each batch of the data and we train the model for 40 epochs (200 iterations).

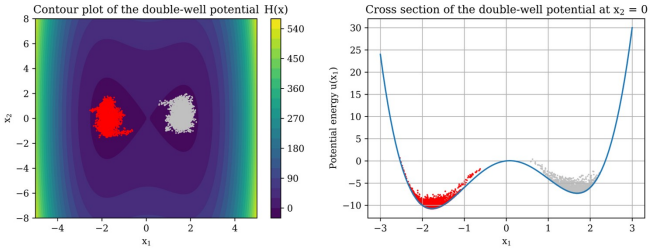

FIG. 3. The samples extracted from Monte Carlo simulations of the double well potential

B. System 2: Müller-Brown potential

The next system of interest we explore in this paper is a particle on the Müller-Brown (MB) potential energy surface. This system is well-studied and is often used to asses reaction path finding algorithm's abilities to find a complex reaction coordinates over a complicated landscape.[21] The MB potential surface is given by:

$$V_{MB}(x,y) = \sum_{i=1}^{4} A_i \exp\left[a_i(x-\bar{x_i})^2 + b_i(x-\bar{x_i})(y-\bar{y_i}) + c_i(y-\hat{y_i})^2\right] \tag{22}$$

The surface can be described as the sum of 4 Gaussians with centers given by the pairs $(\bar{x_j}, \bar{y_j})$. Where the first 3 Gaussians describe the energy minima on the surface and the last Gaussian creates nearly infinite walls around the surface, such that the only low energy regions are described by the first 3 Gaussians. The MB potential parameters are presented in Table I.

TABLE I. MB potential parameters

Parameters	1	2	3	4
a_j	-1	-1	-6.5	0.7
b_j	0	0	11	0.6
c_j	-10	-10	-6.5	0.7
A_j	-200	-100	-170	15
$\hat{x}_j$	1	0	-0.5	-1
$\hat{y}_j$	0	0.5	1.5	1

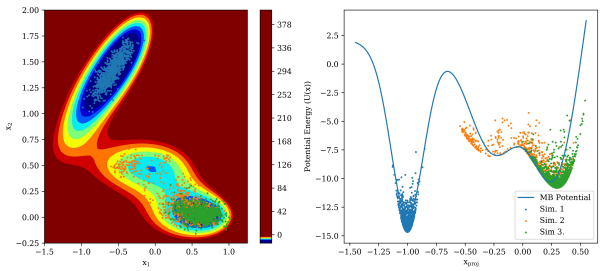

FIG. 4. Visualization of the sampled points on the MB potential. The left figure shows the the samples in the x-y plane. The right figure shows the samples projected onto the $(1,-1)$ vector vs their energy, giving a better idea of the energetic barriers present in the MB potential.

The MB potential represents an increase in complexity from the double-well potential as the potential wells of interest are no longer defined along 1 dimension of the simulation. Three simulations were carried out on the surface of the MB potential, each starting at the centers of the first 3 Gaussians. The first 3 Gaussians represent the metastable states the particle can visit during a simulation. We will refer to the metastable states on this surface as state 1, state 2, and state 3, corresponding to the potentials from left to right, along x_1. All simulations on the MB potential were performed with a $\sigma_{metro} = 0.02$. These simulations were run at a reduced temperature of $k_BT = 1.0$ for 10000 steps with a stride of 10 steps.

Figure 4. shows the samples from the 3 simulations and their corresponding energies projected along the $(1,-1)$ vector. We see that both simulations starting in state 1 and state 3 stay in their respective wells, while the simulation starting in state 2 transitions over the energetic barrier to state 3, resulting in a combination of state 2 and state 3 samples from that simulation. Despite seeing the transition from state 2 to state 3 in our simulation dataset, the MB potential dataset does not have transitions between state 1 and state 2, which will be a region of interest when training Boltzmann generators on this model.

Similar to the double-well potential, we trained multiple Boltzmann generators on the training samples of the MB potential. The Boltzmann generator architecture used for the MB potential consisted of 5 RealNVP blocks, with 2 affine coupling layers per block. Each affine coupling layer contained S and T transformation neural networks with 3 hidden layers with 100 nodes per layer. Hidden layer nodes used the ReLu activation function and the S output layer used the hyperbolic tangent activation function. The training schedule we used for the MB potential is similar to that described in Noé *et al.*[14] An initial model is trained using the ML loss function for 200 iterations, using a mini-batch size of 200, with a learning rate of 0.01. Subsequent Boltzmann generators are initialized with this original ML loss model, and continued training for 600 iterations using a mini-

batch size of 1000, with a learning rate of 0.001. The KL, ML+KL, and ML+KL+RC loss terms were all used to train Boltzmann generators in this fashion.

C. System 3: Dimer in the Lennard-Jones bath

The final system we will explore in this project is the bi-stable dimer in a Lennard-Jones (LJ) bath. The dimer system consists of a 2 particle dimer connected by a 1-D double-well potential (U_{bond} in Equation 23.), surrounded by 36 solvent particles with repulsive LJ potentials ($U_{solvent}$ in 23). The system is contained within a harmonic-box central potential (U_{box} in 23), where particle experience a harmonic potential when they leave the bounds of the box. This model system has 78 dimensions for the x-y coordinates of each of the particles. The dimer simulation represents a simple condensed-matter system, which start to approaches the complexity of full-atom condensed-matter simulations.

With the increase of dimensions of this system, equilibrium samples are much harder to generate for two reasons. Firstly, the 78-D configuration space is exponentially larger than the 2-D systems we've explored previously. Secondly, due to the high density of the system, small changes in the position of particles can result in extremely large energies. The interdependence of particles positions has not been present in the simple models we've explored up to this point.

$$
\begin{aligned}
U(\mathbf{x}) =& U_{bond} + U_{solvent} + U_{box} + U_{center} \\
U_{bond} =& \frac{1}{4}a(d-d_0)^4 - \frac{1}{2}b(d-d_0)^2 + c(d-d_0) \\
U_{solvent} =& \epsilon \sum_{i=1}^{n+1} \sum_{j=i+1, j\neq 2} \left(\frac{\sigma}{||\mathbf{x}_i - \mathbf{x}_j||} \right)^{12} \\
U_{box} =& \sum_{i=1}^{n+2} h(|\mathbf{x}_{ix}| - l_{box}) k_{box} (|\mathbf{x}_{ix}| - l_{box})^2 + \\
& \sum_{i=1}^{n+2} h(|\mathbf{x}_{iy}| - l_{box}) k_{box} (|\mathbf{x}_{iy}| - l_{box})^2 \\
U_{center} =& k_d(\mathbf{x}_{1x} + \mathbf{x}_{2x})^2 + k_d \mathbf{x}_{1y}^2 + k_d \mathbf{x}_{2y}^2
\end{aligned}
\tag{23}
$$

TABLE II. Dimer simulation parameters

Parameter	ϵ	σ	k_d	d_0	a	b	c	l_{box}	k_{box}
Value	1.0	1.1	20	1.5	20.0	10.0	-0.5	3.0	100

Another consideration of the dimer system is the explicit representation of the solvent. As the solvent particles share the same interaction potentials, they are effectively indistinguishable from each other. If the identity of each solvent molecule is preserved over the course of an infinitely long MC simulation, the configuration space of each solvent particle would include the entirety of the box, due to the diffusion and exchange of particles throughout the system. In the short simulations we performed for this system, the particles do not have enough time to visit the entirety of their configuration space. To address the permutational invariance of the solvent particles, we applied the Hungarian algorithm to reassign particles based on their distance to a reference equilibrium sample. This ensures that the x-y coordinates for each particle stays in a local configurational space, despite particles actually exchanging positions throughout the box during the simulation.

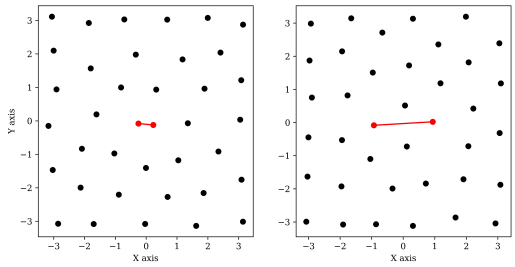

FIG. 5. Visualizations of the compact (left) and extended (right) configurations the dimer simulation can visit. Solvent particles are drawn in black and the dimer particles are drawn in red.

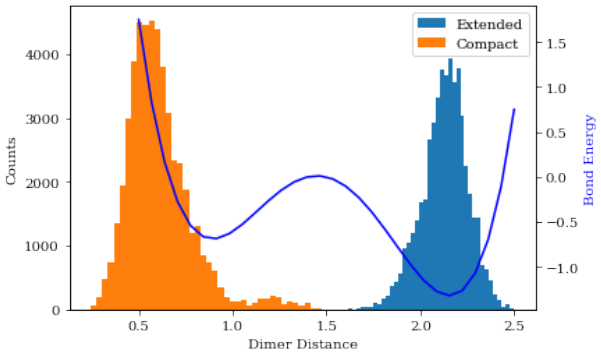

FIG. 6. Distribution of the dimer bond distance from both the extended and compact simulations. The potential energy curve is drawn in blue to show where the compact and extended configurations are energetically favorable. Note, no transitions between the extended and compact configurations are sampled in these simulations.

The dimer system has two metastable states, which we define as the extended and compact configurations shown in Figure 5. Transitions between the extended and compact configurations require significant solvent rearrangement and generally are not observed in the short simulations we performed. To sample this system, we start two simulations in both the extended and compact configuration. Both simulations were run for 50000, with a reduced temperature of $k_BT = 1.0$. In Figure 6., we see that both the compact and extended configura-

tions are well sampled according to the potential energy between the two dimer particles. The 100000 samples generated from these two simulations were used to train Boltzmann generators on this system.

Training Boltzmann generators on the dimer system proved to be more computationally expensive than the simpler systems. We were able to train a Boltzmann generator using the ML loss term. The Boltzmann generator architecture for this system consists of 8 RealNVP blocks, with 2 affine coupling layers per block. Each affine coupling layer contains S and T transform networks with 3 hidden layers and 200 nodes per layer. Hidden layer nodes use the ReLu activation function and the S output layer uses the hyperbolic tangent activation function. The training schedule for the ML loss Boltzmann generator trained for 250 iterations uses a mini-batch size of 8000 samples, with a learning rate of 0.0001. Other loss functions are too computationally expensive to generate results for in a reasonable amount of time.

IV. RESULTS AND DISCUSSIONS

A. System 1: Double well Potential

Given the fact that the ML loss helps the Boltzmann generator to focus on relevant parts of state space and is therefore useful in the early stages of training, for this simple system, we start with training the model by examples (on the ML loss). Then, based the model trained on the ML loss (ML model), we train the model on the following three combinations of loss functions:

- KL model: train the ML model on the KL loss for another 200 iterations. $(w_{ML},\ w_{KL},\ w_{RC}) = (0,\ 1,\ 0)$.
- ML + KL model: train the ML model on the ML loss and KL loss simultaneously for another 200 iterations. $(w_{ML},\ w_{KL},\ w_{RC}) = (1,\ 1,\ 0)$.

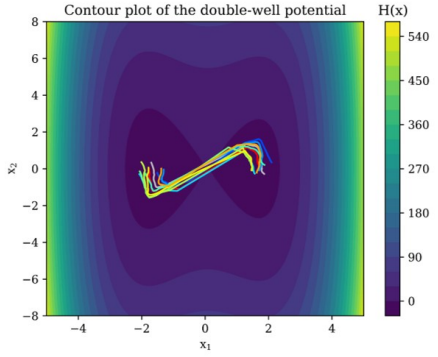

FIG. 7. The linear interpolation in latent space with the RC model for the double well potential model

- RC model: train the ML model on the all kinds of loss functions (ML, KL, and RC) simultaneously for another 200 iterations. $(w_{ML},\ w_{KL},\ w_{RC}) = (1,\ 1,\ 1)$.

As a result, the loss functions of all of the models converge within 200 iterations (see Double-well potential.ipynb). As shown in Figure 8., the top row of Figures shows how the samples in configuration space are mapped to latent space. The samples correspond to the left well in the configuration space are colored in red, while the ones corresponding to the right well are colored in silver (see Figure 3.). From the top row of Figures, we can observe that the ML loss function does help the model sample both energy minima. On the other hand, focusing on the most stable metastable state (which is the well on the left), the KL model maps samples in the well on the left to the center of the prior Gaussian distribution, while the rest of the samples are mapped to a low-probability region. Therefore, when converting random samples drawn from the Gaussian distribution to configuration space, the KL model generates much fewer samples in the right well, but more samples in the left well compared with the ML model. Predictably, as an intermediate between the ML model and the KL model, the ML + KL model combines the strengths of the two such that it generates more configurations of the lowest energies, but also generates a good amount of samples in the well on the right at the same time. RC model, which trains on the RC loss with RC defined as x_2, exhibits similar behaviors as the ML + KL model.

With a trained Boltzmann generator, we perform linear latent interpolation with each model to seek physical interpretations of direct paths in latent space. As a result, as the straight pathways in latent space are mapped to configuration space, they turn out to be nonlinear pathways that correspond to transition region having low energy (hence high probability) between the metastable states, which are in agreement with the results presented in the original paper.[14] Although reaction pathways in practice are not necessarily the paths with the lowest energy, these predicted paths still provide decent candidates of reaction coordinates for bias-enhanced or path-based sampling methods. (For the latent interpolation results of other models, please refer to Double-well potential.ipynb.)

Lastly, given that the Boltzmann generators trained on different loss functions are able to generate statistically independent configuration samples, we combine them with either weighted histograms or batchwise kernel density estimation. Figure 9. shows the free energy as a function of x_2, which is the reaction coordinate of the double-well potential, predicted by the combination of KDE with different models. As can be seen in the Figure, while there is a discrepancy between the predicted curve and the analytical solution around the saddle point of the free energy surface (which is a region of high energy), each model has decent performance at

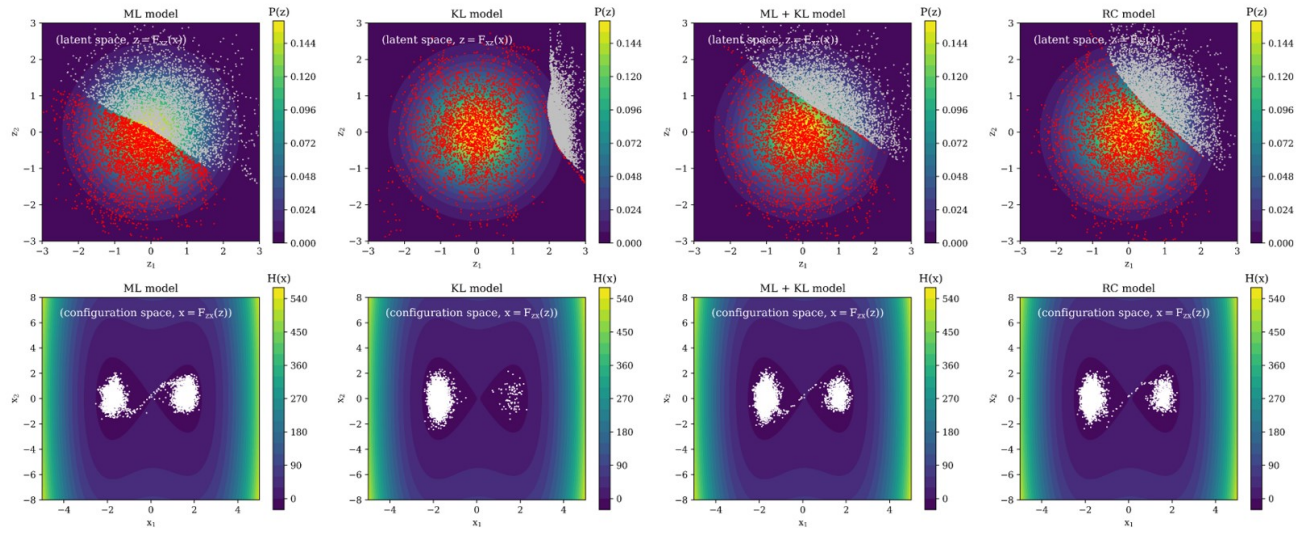

FIG. 8. Samples generated in latent space (the top row) and configuration sapce (the bottom row) by different models

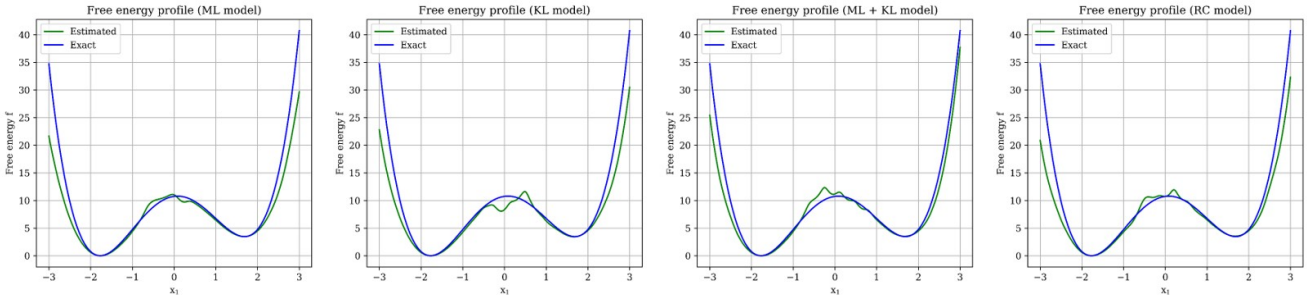

FIG. 9. The free energy profile of the double well potential model as a function of the selected reaction coordinate (x_2) predicted by the combination of KDE with different models.

generating low-energy configurations. Given that the data points of high-energy configurations are not reported in free energy profiles in the original paper, our models well agree with the ones implemented in the original paper. In addition, the perfect overlap between the prediction and the analytical solution at both energy minima enables accurate computation of the free energy difference between two metastable states, showing the strength of Boltzmann generators of avoiding the usage of alchemical intermediate states or the knowledge of the predefined reaction coordinate compared with other advanced sampling methods in free energy calculations.

B. System 2: Müller-Brown potential

Boltzmann generators trained on the MB potential samples are easily able to recreate the phase-space distribution. We see good agreement between the generated points and the underlying potential energy for all models in Figure 11. Comparing the ML model to the KL model, we see much better sampling along the low energy path between the state 1 and state 3 potential wells. This can be attributed to the KL loss's penalization for high energy configurations. This forced the samples that the ML loss Boltzmann generator placed between the systems to find a lower energy path when

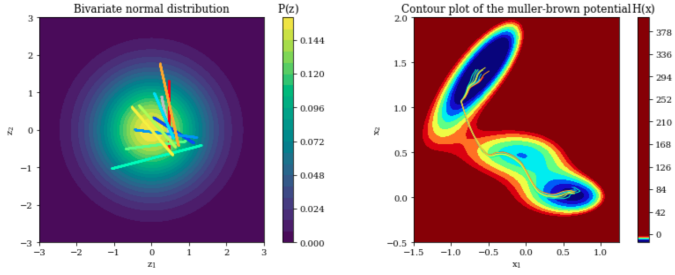

FIG. 10. Interpolation of points from the MB potential states 1 and 3 in the latent space, transformed into the configuration space.

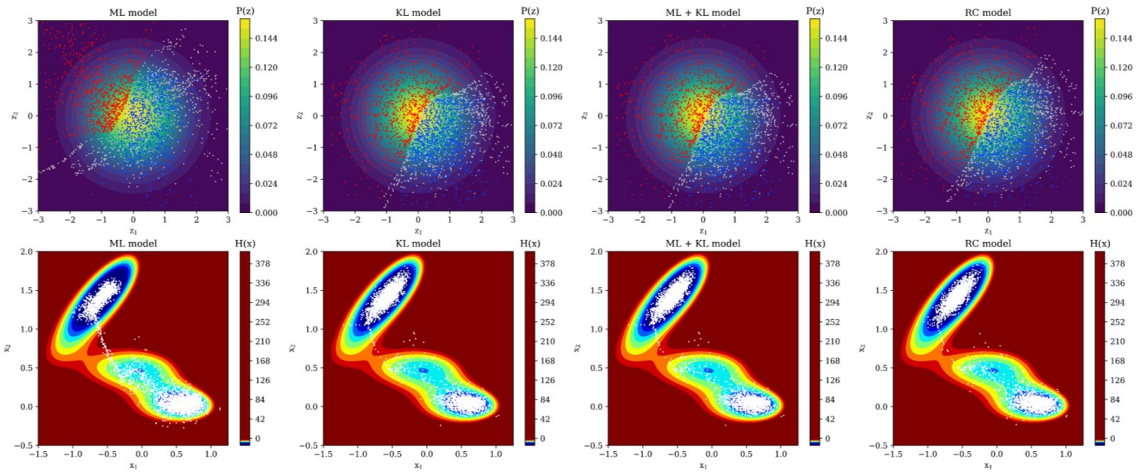

FIG. 11. Resulting distributions in the latent and configuration space for the MB potential. Training samples transformed to the latent space from state 1, state 2 and state 3 are colored red, gray and blue respectively. White points represent generated samples from the respective Boltzmann generator models.

further trained on the KL loss Boltzmann generator.

As the KL model already had samples near the minimum energy transition path between state 1 and state 2, the ML+KL and RC loss models saw no change in the placement of samples between states 1 and 2. However, looking at the free energy profile of the RC loss model, shown in Figure 12., we see a better approximation of the large free energy barrier between state 1 and state 2, due in part to slightly better sampling in that low probability region. Noé *et al.* showed that the RC loss term greatly contributed to sampling the transition between regions, therefore we suspect there may be an error in our current implementation of the RC loss term.

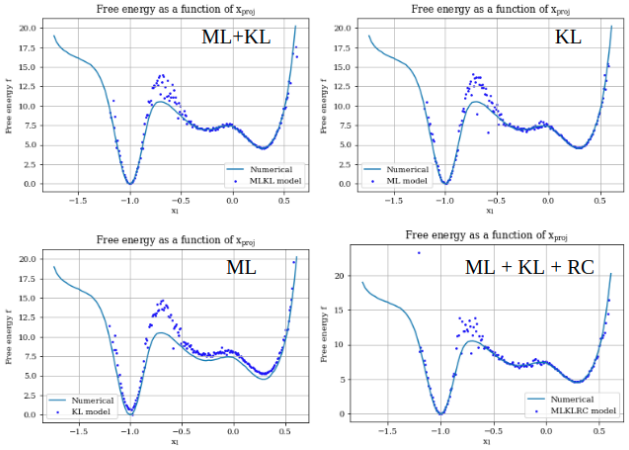

FIG. 12. Free energy approximation from various Boltzmann generator model types for the MB potential.

By interpolating between samples from state 1 and state 2, we see that the Boltzmann generators are able to propose reasonable reaction paths over the complex MB potential surface. In Figure 10., we see a collapse of interpolated lines in the latent space to a single reaction coordinate in the configuration space, similar to what was observed in the original paper. With just samples from the individual metastable states, Boltzmann generators were able to estimate the reaction coordinate between all 3 states on the MB potential surface.

C. System 3: Dimer in the Lennard-Jones bath

Despite only being able to train an ML loss Boltzmann generator on the dimer simulation, we were able to generate reasonable samples by visual inspection. In Figure 13., we see that the Boltzmann generator is able to learn the placement of solvent particles about the dimer. Some particle phase spaces remained localized to one area, shown by the average particles (in black and red) at the center of their distribution (gray), of the simulation. However the solvent particles that rearranged to allow the dimer to transition between the extended and compact distributions are not as localized.

Figure 14. shows that the generated configurations sample the intermediate states between the extended and compact configurations. However, when comparing the energy of the generated configurations, we see a large discrepancy between the energies from the simulation energies and the generated energies, shown in Figure 15. The ML loss model is effective at matching the configuration space distribution, but does not directly use the energy ($u(\mathbf{x})$) function to distinguish favorable and unfavorable configurations.

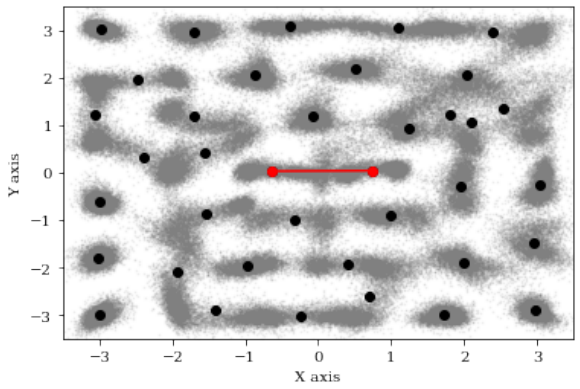

FIG. 13. Visualization of the generated dimer system distribution. Average solvent and dimer positions are drawn in black and red respectively. All generated particle positions are plotted in gray.

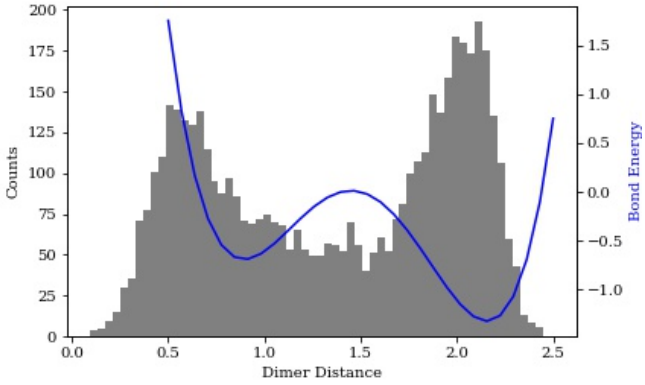

FIG. 14. Generated distribution of bond distances (gray) for the dimer system using the ML loss function. The potential energy along the bond (blue) shows the favorable distances of the dimer particles.

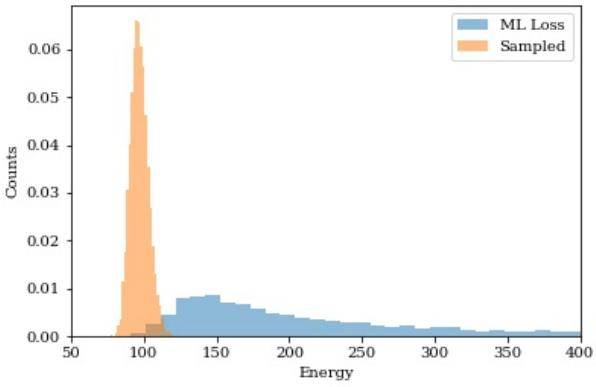

FIG. 15. Comparison of the energy distributions of the simulated and generated configurations of the dimer system.

Due to the large differences in energy between the simulated and generated configurations, when applying the free energy calculation we applied to the previous systems, the statistical weights for all generated samples were zero, meaning the configurations generated were too far from the true Boltzmann distribution to use reweighting methods. Unfortunately, when applying more complex Boltzmann generators to the dimer system, such as the KL loss model, the calculation of computing the gradient of the dimer system's energy proved to be computationally intractable with our implementation of the energy function. When revisiting this project, more care will be needed when programming the energy function, such that the gradient of the energy is computed as efficiently as possible. We also explored using PyTorch's GPU acceleration for handling the gradient calculation with little success.

V. CONCLUSION

In this study, we were able to apply a variety of Boltzmann generators to 3 toy model systems. Using the samples generated from these Boltzmann generators we were able to generate statistically independent samples equilibrium from multiple metastable stables separated by large energetic barriers in one shot. Thereby, addressing the rare event-sampling problem many simulations are plagued with using deep learning algorithms. Using Boltzmann generators we were also able to estimate free energies to a high degree of accuracy for the double-well potential and MB potential systems. Using latent space interpolation, we could also generate reaction coordinates on both potential energy surfaces.

Unfortunately, the Boltzmann generator applied to the dimer system was not able to generate samples close enough to the Boltzmann distribution to estimate free energies between states. However, we were still able to apply a Boltzmann generator to a much higher dimensional system than the first two potential energy surfaces we explored in this paper, showing that it is possible to train Boltzmann generators on larger dimensional spaces. In Noé *et al.* original paper, Boltzmann generators were used to estimate the free energy of a conformational switch in a BPTI protein.[14] Boltzmann generators represent a novel way of addressing rare event-sampling in many-body systems for less computational cost than traditional enhanced sampling methods.

The challenging part of applying any deep learning technique to a problem is choosing an appropriate representation of the data that will capture the features of the simulation we're interested in studying. In the dimer simulation, we saw a glimpse of this, as we removed the permutational invariance of the solvent through the relabeling of the solvent particles. For protein systems, deep learning methods often represent the various configurations of protein using a combination of torsional degrees of freedom and atom cartesian coordinates.[14,22]

Boltzmann generators, a flow-based generative model, are especially promising in the field of molecular simulations due to a high degree of scalability. Typical applications of RealNVP networks in deep learning include celebrity face generation[23] and raw audio waveform generation[24] which can address problems with 10^6 dimensions. Granted we have samples of states of interest, Boltzmann generators can be a useful tool for estimating free energy differences in protein-ligand binding complexes, protein folding dynamics, and macrologies phase changes, as many problems can be framed similarly to the problems presented in this project.

VI. FUTURE APPLICATIONS

A. Exploration of the configurational space of flexible molecules

The sampling of the configurational space of flexible molecules, such as an unfolded protein, or a protein-nucleic acid binding complex, has been one of most challenging problems in the field, not only due to the rough free energy surface and high energy barriers between the metastable states (e.g. the bound state and the unbound state of a binding complex), but also the considerably large phase space of the system as a result of significant flexibility of the molecules. The curse of dimensionality exacerbates the problem of selecting reaction coordinates that are sufficient to comprehensively capture all the slow degrees of freedom. In addition, investigation with sampling methods taking advantages of alchemical intermediate states is not even easier, since the scaling of the electrostatic interactions between the ligand and the receptor might lead to the so-called greasy core problem, i.e. after the electrostatic interactions are decoupled, the ligand becomes kinetically trapped because it is too hydrophobic to get into the binding cavity which is full of water molecules.

In this situation, Boltzmann generators could be a promising approach on account of its strengths of not relying on predetermined reaction coordinates or alchemical intermediate states. For example, the C-terminal SH2 domain of phospholipase C-γ1 (PLCC) is known for its ability to bind to particularly diverse peptides that deviate from its specificity profile. This structure along with similar species have been received increasing research attention[25,26] in their binding mechanisms of dissimilar ligands. In this situation, if Boltzmann generators are able to generate statistically independent configurations from both the bound state and the unbound states given a reasonable training dataset, we can not only accurately compute the binding free energy, but also hopefully determine the binding ensemble of the binding complex. If the binding ensemble is correctly predicted, we can examine whether PLCC is able to adopt alternate binding conformations which were not predicted based on the available crystal structures and the consensus sequence obtained experimentally. From the most representative binding structure, we are also likely to get more insight into the binding mechanism of PLCC and explain its multimodal specificity.

B. Estimating heteropolymer secondary structure

Applying Boltzmann generators to macro-molecule folding is another possible avenue for novel research. Many research groups are studying and designing non-natural heteropolymers.[27,28,29] These macromolecules behave similarly to proteins, however can be synthetically designed to have much more functional diversity. A big effort in the field of heteropolymers is to design the secondary structure given a proposed monomer. Currently the design of novel heteropolymers relies heavily on chemical intuition and requires a great deal of experimental trial and error. Computational structure prediction methods for short N-mers of a novel heteropolymer is not a trivial task. Applying a Boltzmann generator with adaptive sampling to the internal coordinates of a proposed heteropolymer N-mer, could be an efficient way to quickly explore and generate ensembles of folded and unfolded configurations.

REFERENCES

[1] David B Wells and Aleksei Aksimentiev. "Mechanical properties of a complete microtubule revealed through molecular dynamics simulation". In: *Biophysical journal* 99.2 (2010), pp. 629–637. DOI: 10.1016/j.bpj.2010.04.038..

[2] Veronica Salmaso and Stefano Moro. "Bridging molecular docking to molecular dynamics in exploring ligand-protein recognition process: An overview". In: *Frontiers in pharmacology* 9 (2018), p. 923. DOI: 10.3389/fphar.2018.00923.

[3] Wasinee Khuntawee et al. "Molecular dynamics study of natural rubber–fullerene composites: connecting microscopic properties to macroscopic behavior". In: *Physical Chemistry Chemical Physics* 21.35 (2019), pp. 19403–19413. DOI: 10.1039/C9CP03155C.

[4] Alessandro Laio and Michele Parrinello. "Escaping free-energy minima". In: *Proceedings of the National Academy of Sciences* 99.20 (2002), pp. 12562–12566. DOI: 10.1073/pnas.202427399.

[5] Glenn M Torrie and John P Valleau. "Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling". In: *Journal of Computational Physics* 23.2 (1977), pp. 187–199. DOI: 10.1016/0021-9991(77)90121-8.

[6] Eric Darve, David Rodrıguez-Gómez, and Andrew Pohorille. "Adaptive biasing force method for scalar and vector free energy calculations". In: *The Journal of chemical physics* 128.14 (2008), p. 144120. DOI: 10.1063/1.2829861.

[7] Yuji Sugita, Akio Kitao, and Yuko Okamoto. "Multidimensional replica-exchange method for free-energy calculations". In: *The Journal of Chemical Physics* 113.15 (2000), pp. 6042–6051. DOI: 10.1063/1.1308516.

[8] Vittorio Limongelli, Massimiliano Bonomi, and Michele Parrinello. "Funnel metadynamics as accurate binding free-energy method". In: *Proceedings of the National Academy of Sciences* 110.16 (2013), pp. 6358–6363. DOI: 10.1073/pnas.1303186110.

[9] Robert Swendsen and Jian-Sheng Wang. "Replica Monte Carlo Simulation of Spin-Glasses". In: *Physical review letters* 57 (1986), pp. 2607–2609. DOI: 10.1103/PhysRevLett.57.2607.

[10] Yuji Sugita and Yuko Okamoto. "Replica-exchange molecular dynamics method for protein folding". In: *Chemical physics letters* 314.1-2 (1999), pp. 141–151. DOI: 10.1016/S0009-2614(99)01123-9.

[11] AP Lyubartsev et al. "New approach to Monte Carlo calculation of the free energy: Method of expanded ensembles". In: *The Journal of chemical physics* 96.3 (1992), pp. 1776–1783. DOI: 10.1063/1.462133.

[12] Alejandro Gil-Ley and Giovanni Bussi. "Enhanced conformational sampling using replica exchange with collective-variable tempering". In: *Journal of chemical theory and computation* 11.3 (2015), pp. 1077–1085. DOI: 10.1021/ct5009087.

[13] Hiraku Oshima, Suyong Re, and Yuji Sugita. "Replica-exchange umbrella sampling combined with Gaussian accelerated molecular dynamics for free-energy calculation of biomolecules". In: *Journal of chemical theory and computation* (2019). DOI: 10.1021/acs.jctc.9b00761.

[14] Frank Noé et al. "Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning". In: *Science* 365.6457 (Sept. 6, 2019). DOI: 10.1126/science.aaw1147.

[15] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using real NVP". In: *arXiv preprint arXiv:1605.08803* (2016).

[16] Emanuel Parzen. "On estimation of a probability density function and mode". In: *The annals of mathematical statistics* 33.3 (1962), pp. 1065–1076.

[17] Richard A Davis, Keh-Shin Lii, and Dimitris N Politis. "Remarks on some nonparametric estimates of a density function". In: *Selected Works of Murray Rosenblatt*. Springer, 2011, pp. 95–100.

[18] P.W.D. Charles. *Project Title*. https://github.com/charlespwd/project-title. 2013.

[19] Martın Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: http://tensorflow.org/.

[20] Adam Paszke et al. "PyTorch: An imperative style, high-performance deep learning library". In: *Advances in Neural Information Processing Systems*. 2019, pp. 8024–8035.

[21] Silvia Bonfanti and Walter Kob. "Methods to locate Saddle Points in Complex Landscapes". In: *The Journal of Chemical Physics* 147.20 (Nov. 28, 2017), p. 204104. ISSN: 0021-9606, 1089-7690. DOI: 10.1063/1.5012271. arXiv: 1701.01241. URL: http://arxiv.org/abs/1701.01241 (visited on 05/04/2020).

[22] Huang-Wei Chang et al. "Persistent Topology and Metastable State in Conformational Dynamics". In: *PLOS ONE* 8.4 (Apr. 2, 2013), e58699. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0058699.

[23] Diederik P. Kingma and Prafulla Dhariwal. "Glow: Generative Flow with Invertible 1x1 Convolutions". In: *arXiv:1807.03039 [cs, stat]* (July 10, 2018). arXiv: 1807.03039.

[24] Aaron van den Oord et al. "WaveNet: A Generative Model for Raw Audio". In: *arXiv:1609.03499 [cs]* (Sept. 19, 2016). arXiv: 1609.03499.

[25] Marissa A McKercher et al. "Multimodal recognition of diverse peptides by the C-terminal SH2 domain of phospholipase C-γ1 protein". In: *Biochemistry* 56.16 (2017), pp. 2225–2237.

[26] Marissa A McKercher et al. "Diversity in peptide recognition by the SH2 domain of SH2B1". In: *Proteins: Structure, Function, and Bioinformatics* 86.2 (2018), pp. 164–176.

[27] Krzysztof Ziach et al. "Single helically folded aromatic oligoamides that mimic the charge surface of double-stranded B-DNA". In: *Nature Chemistry* 10.5 (May 2018), pp. 511–518. ISSN: 1755-4349. DOI: 10.1038/s41557-018-0018-7.

[28] Richard P. Cheng, Samuel H. Gellman, and William F. DeGrado. "-Peptides: From Structure to Function". In: *Chemical Reviews* 101.10 (Oct. 1, 2001), pp. 3219–3232. ISSN: 0009-2665. DOI: 10.1021/cr000045i.

[29] Samuel H. Gellman. "Foldamers: A Manifesto". In: *Accounts of Chemical Research* 31.4 (Apr. 1, 1998), pp. 173–180. ISSN: 0001-4842. DOI: 10.1021/ar960298r.