Using VAEs to Learn Latent Variables: Observations on Applications in cryo-EM

Edelberg, Daniel G.
Yale University

Lederman, Roy R.
Yale University

Technical Report YALEU/TR-1565
January 2023

Abstract

Variational autoencoders (VAEs) are a popular generative models used to approximate distributions. The encoder part of the VAEs is used in amortized learning of latent variables, producing a latent representation for the data samples. Recently, VAEs have been used to characterize physical and biological systems. In this case study, we qualitatively examine the amortization properties of a VAE used in biological applications. We find that in this application the encoder bears a qualitative resemblance to more traditional explicit representation of latent variables.

1 Introduction

Variational Autoencoders (VAEs) provide a deep learning method for efficient approximate inference for problems with continuous latent variables. A brief reminder about VAEs is presented in Section 2.1; a more complete description can be found, inter alia, in [1, 2, 3, 4, 5, 6]. Since their introduction, VAEs have found success in a wide variety of fields. Recently, they have been used in scientific applications and physical systems [7, 8, 9, 10, 11].

Given a set of data $x = \{x_i\}$ VAEs simultaneously learn an encoder Enc_ξ that expresses a conditional distribution $q_\xi(z|x)$ of a latent variable z_i given a sample x_i, and a decoder Dec_θ which expresses the conditional distribution $p_\theta(x|z)$ given the latent variable. They are trained using empirical samples to approximate the distribution $p_\theta(x, z)$.

In this work we focus on the properties of the encoder $q_\xi(z|x)$ that arise as an approximation of the distribution $p_\theta(z|x)$. A single encoder $q_\xi(z|x)$ is optimized to be able to produce the distribution of latent variable z for any input x, which is a form of amortization. Intuitively, one might expect that the encoder $q_\xi(z|x)$ would generalize well to plausible inputs that it has not encountered during the optimization/training procedure. Indeed, this generalization is observed in many applications, and the ability of the encoder to compute the latent variables for new unseen data points is used in some applications. In addition, the variational construction sidesteps a statistical problem by marginalizing over the latent variables to approximate the maximum-likelihood estimator (MLE) for some parameters θ of the distribution $p_\theta(x, z)$, rather than θ *and* the latent variables z_i associated with each sample x_i; in the latter case, the number of variables grows together with the number of samples and the estimates of $p_\theta(x, z)$ may not converge to the true solution.

We present a qualitative case study of the amortization in VAEs in a physical problem, looking at a VAE applied to the problem of continuous heterogeneity in cryo-electron microscopy (cryo-EM), implemented in CryoDRGN [7]. We examine the hypothesis that the encoder in this VAE generalizes well to previously unseen data, and we compare the use of a VAE to the use of an explicit variational estimation of the distribution of the latent variables. In order to study the generalization in a realistic environment, we exploit well-known invariances and approximate invariances in cryo-EM data to produce natural tests.

Our case study suggests that in this case the encoder does not seem to generalize well; this can arguably be interpreted as a form of overfitting of the data. Furthermore, we find that using explicit latent variables (variational approximations and arguably also explicit values) yields qualitatively similar (and arguably better) estimates than using the encoder in this test case.

We would like to clarify that the purpose of this case study is not to criticize the work in CryoDRGN [7], but rather to draw attention to possibly surprising properties of VAEs in some applications and to the parallels between VAEs and explicit latent variables in these applications. The phenomena observed here are not exhibited in any application; for completeness we present in Appendix B similar experiments on classic MNIST VAEs.

The work is loosely inspired by the work in [12]; one could conceivably draw some conceptual parallels between the over-fitting demonstrated there and some of the experiments presented in this paper in the context of VAEs.

The code used for this paper will be made available at

`https://github.com/danieledelberg/ExplicitLatentVariables`.

2 Preliminaries

2.1 Variational Inference and Variational Autoencoders

In this section we provide a brief reminder of VAEs, adapted from the formulation in [2].

Figure 1 illustrates the VAE neural network, combining an encoder Enc_ξ and a decoder Dec_θ. For the observations $\{x_1, \ldots, x_n\}$, we aim to infer a distribution $p_\theta(x)$. We denote by $z_1, \ldots, z_n$ the latent variables; these are an unobserved component of the model. The parameters of the generative model for the decoder are denoted by θ. The marginal distribution $p_\theta(x)$ over the observed variables, is given by

$$p_\theta(x) = \int_z p_\theta(x, z) dz \tag{1}$$

The classic Maximum-Likelihood Estimation problem gives the following optimization problem to solve:

$$\theta^* = \underset{\theta}{\text{argmax}}\, L(x, \theta) = \underset{\theta}{\text{argmax}} \sum_{i=1}^{n} \log p(x_i|\theta) \tag{2}$$

Or equivalently,

$$\theta^* = \underset{\theta}{\text{argmax}} \sum_{i=1}^{n} \log \int_{z_i} p_\theta(x|z) p_\theta(z) dz \tag{3}$$

However, this integration in (3) is intractable, and thus it is desirable to have an approximation to the distribution $p_\theta(z|x)$. We utilize an encoder model $q_\xi(z|x)$, where ξ are the parameters of this inference model, called the variational parameters. We intend to optimize these parameters to best approximate

$$q_\xi(z|x) \approx p_\theta(z|x) \tag{4}$$

The distribution $q_\xi(z|x)$ can be parameterized by a neural network. A popular choice is a neural network encoder $\text{Enc}_\xi(x_i)$ which produces a mean μ_i and variance σ_i of a multivariate Gaussian distribution for any input x_i,

$$\text{Enc}_\xi(x_i) = (\mu_i, \log \sigma_i) \tag{5}$$

$$q_\xi(z_i|x_i) = \mathcal{N}(z_i; \mu_i, \text{diag}\,(\sigma_i)) \tag{6}$$

Since the function Enc_ξ shares its variational parameters ξ across all x_i's, this process is called amortized variational inference. This is in contrast to fitting explicit distributions $q_{\xi_1}(z_1|x_1), \ldots, q_{\xi_n}(z_n|x_n)$ for each

datapoint separately with parameters $\xi_1, \ldots, \xi_n$ that are not shared across distributions (the latter is what we will later do in the experiment in Section 3.2). In many applications, the amortized learning of shared encoder variable reduces the computational cost; the encoder often generalizes so that it can produce a reasonable approximate $q_\xi(z|\hat{x})$ for samples $\hat{x}$ that were not used in training.

In the decoder, the parameters θ are fitted to approximate the true distribution $p^*(x|z)$ in a similar manner as we described for the case of the encoder. In order to fit the distribution $q_\xi(z|x)$, one must formulate an alternative objective function, since the original maximum-likelihood formulation in (2) does not include a distribution q. We write the objective function as:

$$L(x, \theta) = \mathbb{E}_{q_\xi(z|x)} \left[\log p_\theta(x, z) - \log q_\xi(z|x)\right] + \mathbb{E}_{q_\xi(z|x)} \left[\log q_\xi(z|x) - \log p_\theta(z|x)\right] \tag{7}$$

The second expectation term is the Kullback-Leibler (KL) divergence between $q_\xi(z|x)$ and $p_\theta(z|x)$, written as $D_{KL}(q_\xi(z|x)\|p_\theta(z|x))$, which is non-negative. The first expectation term is called the evidence-based lower bound (ELBO) $\mathcal{L}_{\theta,\xi}(x)$. The ELBO loss serves as a lower-bound to $L(x, \theta)$ and we attempt to find a series of distributions $q_1, \ldots, q_n$ that will maximize this lower bound. For completeness, a reminder of the derivation of the ELBO loss, is presented in Appendix A. When we use the ELBO loss as an objective function for optimization, we often write it as

$$\mathcal{L}_{\theta,\xi}(x) = \mathbb{E}_{q_\xi(z|x)} \left[\log p_\theta(x|z)\right] - D_{KL}(q_\xi(z|x)\|p_\theta(z)) \tag{8}$$

Typically, we choose $q_\xi(z|x)$ to be a multivariate Gaussian distribution and $p_\theta(z) \sim \mathcal{N}(0, 1)$ so that the KL Divergence term can be written analytically. The first term on the RHS is used as the expected reconstruction error.

The amortization gap [4] characterizes the difference between the optimized $q_{\hat{\eta}}$ and the best possible posterior from the set of all possible distributions parameterized by the network Enc_ξ with respect to the ELBO.

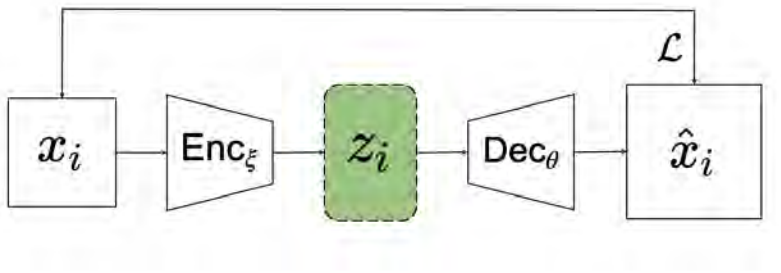


Figure 1: Experimental setup of a standard VAE. x_i is a datapoint used as input, z_i is the associated latent space point, $\hat{x}_i$ is the reconstructed output of x_i. $\mathcal{L}$ represents the calculation of the ELBO function between x and $\hat{x}_i$

2.2 Case Study: Cryo-Electron Microscopy

Cryo-EM is an imaging technology that uses an electron beam to create a tomographic projection of a biological macromolecule frozen in a layer of vitreous ice. The location and orientation of particles within the ice is random and unknown. The aim of classic cryo-EM reconstruction is to estimate the 3D structure of the biomolecule using the 2D particle images. Due to the radiation sensitivity of the biomolecules and lack of contrast enhancement agents, particle images often have a very low signal-to-noise (SNR) ratio. Advances in recent years have facilitated reconstruction of these 3D structures at resolutions approaching 1-2Å[13]. One of the actively studied questions in cryo-EM is the question of heterogeneity: how to infer multiple 3D structures (molecular conformations) from measurements of mixtures of different macro-molecules. Despite the success in analyzing mixtures of distinct discrete conformations, the analysis of continuums of conformations (e.g. flexible molecules) has been an open problem until recent years, and it remains an important active area of

research [14]. A very brief formulation of the problem is presented in the next subsection. A more detailed description of the cryo-EM problem formulation can be found, inter alia, in [15].

2.2.1 Mathematical Formulation and Image Model

Let $V : \mathbb{R}^3 \to \mathbb{R}$ be the 3D structure we want to estimate. In the simplified cryo-EM model, we assume that the particle image x_i is created by rotating V by some rotation $\omega_i \in SO(3)$, performing a tomographic projection, shifting by some $t_i \in \mathbb{R}^2$, convolving with the contrast transfer function of the microscope h_i, and adding noise. This can be written as

$$x_i = h_i * T_{t_i} P R_{\omega_i} V + \epsilon_i \tag{9}$$

where R_{ω_i} is the rotation operator that applies rotation ω_i to the volume, T_{t_i} is the equivalent operator for the translation, P is the tomographic projection operator, and ϵ_i is the frequency-dependent noise. In the heterogeneous problem, the single volume V is replaced with a function V_z, where z is the latent conformation variable, which is often embedded in a lower-dimensional space in the continuous formulation introduced in [16, 17]. In the discrete formulation of the heterogeneous problem, $z = \{1, \ldots, K\}$ represents the K discrete volumes of the structure [15].

2.2.2 Case Study VAE: CryoDRGN

One of the methods for analyzing continuous heterogeneity in cryo-EM is CryoDRGN, a customized VAE specifically designed for generating 3D biomolecule structures from pre-processed micrographs [7, 18]. The network utilizes an initial pose set, composed of rotations $\omega_1, \ldots, \omega_n$, translations $t_1, \ldots, t_n$, and contrast transfer functions $h_1, \ldots, h_n$, provided by an upstream homogeneous reconstruction, typically via an expectation-maximization algorithm. CryoDRGN uses an encoder Enc_ξ to estimate the conformation variable z, then a specialized neural network-parameterized decoder to render a slice $\hat{x}_i$ of volume V_i for each image x_i and latent variable z_i. This is illustrated in Figure 2.

Our experiments are based on the original CryoDRGN network, with modification only to the encoder component of the network.

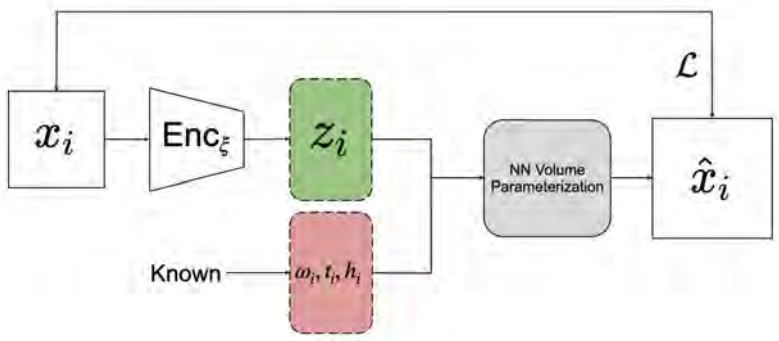


Figure 2: The CryoDRGN Pipeline for heterogeneous reconstruction. x_i is the cryo-EM image datapoint as input. ω_i, t_i, h_i are the given rotation, translation, and contrast transfer function of the input image. These variables with latent variable z_i are used as input to the CryoDRGN volume decoder which then generates a projection image $\hat{x}_i$, which is compared to x_i via the $\mathcal{L}$ loss function and parameters are updated accordingly.

3 Methods and Results

Our experiments are based on modified versions of the CryoDRGN code; the original code is available at `https://github.com/zhonge/cryodrgn`. In order to make the comparison as informative as possible, we

tried to minimize the modifications to the original code, rather than optimize the alternative setups. For similar reasons, we chose a well-studied dataset with well-studied parameters used in the CryoDRGN tutorial [19] as discussed in Section 3.1.

Our naive null assumption is that the encoder utilizes the structure of the data to produce an informative latent space; we introduce a number of modifications to qualitatively test this hypothesis. In Section 3.1 we establish a baseline for our experiments by following a given tutorial of the CryoDRGN software package. In Section 3.2 we compare qualitatively the amortized encoder to a simple implementation of analogous explicit variational approximations of latent variables associated with each particle image separately. In Section 3.3 we investigate qualitatively the ability of the autoencoder to work without real information content. In Section 3.4 we qualitatively test the ability of the network to generalize to small perturbations in the data.

3.1 Baseline

As a baseline for our case-study we used the well-studied EMPIAR-10076 L17-Depleted 50S Ribosomal Intermediates dataset [20] with well-studied parameters for CryoDRGN, used in the CryoDRGN tutorial described in detail in [19]. We chose to focus on the first part of the tutorial, where the particle images are downsampled to 128 x 128 and run through the initial training process. As mentioned above, CryoDRGN uses estimates for the rotation, translation, and contrast transfer function for each particle that are provided by upstream algorithms, and focuses on the conformation variable z.

The results of the baseline experiment are illustrated in Figure 3. We note that different initialization of the algorithm leads to results that are different, but qualitatively comparable. The scatter plot represents the means of the latent variable z_i for each particle image; the UMAP algorithm is used for two-dimensional visualization of the 8-dimensional latent variable. We use the latent variables associated with the first ten images as reference points (in red). In addition, we present some examples of 3-D conformation that the decoder produces from the latent variables of the first six particle images.

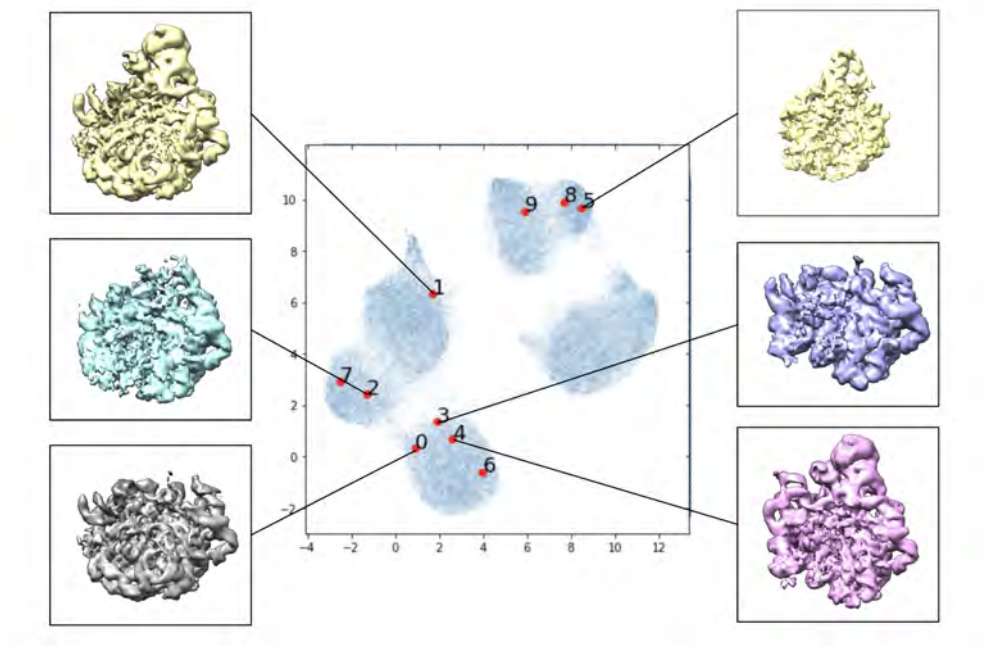


Figure 3: Low-resolution preliminary CryoDRGN results following their tutorial. Labeled locations in the learned latent space of the first ten images in the data set visualized with UMAP dimensionality reduction. Volumes associated with certain areas of the latent space are included.

3.2 Experimental System: Variational Lookup Table

In this section we qualitatively compare the amortized encoder to a simple implementation of an analogous explicit variational approximations of latent variables associated with each particle image separately. We replace the encoder deep network $\text{Enc}_\xi : x_i \to (\mu_i, \sigma_i)$ from Section 2.1 with a lookup table $\text{LT}_i : i \to (\mu_i, \sigma_i)$, to which we refer as a Variational Lookup Table (VLT). The VLT keeps track of an explicit mode μ_i and variance σ_i for the latent variable for each individual image separately from the other images, whereas the encoder keeps a set of parameters ξ that is shared across all the images.

In a classic VAE or in the CryoDRGN architecture, an index i is used to choose a point from the training set x_i to feed-forward through the encoder, producing a parameterization of $q_\xi(z_i, x_i)$. The VLT chooses a row in a table associated with the index, which corresponds to a parameterization of the latent distribution q for the i-th particle image. The architecture is implemented by replacing the encoder model in the CryoDRGN software package with a Pytorch embedding table. In both architectures, during the optimization process, a value $z \sim q$ is sampled from q for each particle image, and passed through the decoder. The entries in the table are optimized by traditional backpropagation methods. The table can also be optimized using a separate optimization scheme or learning rate from the decoder.

This architecture shares some features with the tools described in [21], although our implementation allowed for some additional experimentation with optimization methods and has been adjusted to fit within the CryoDRGN pipeline. The architecture of the VLT is illustrated in 4.

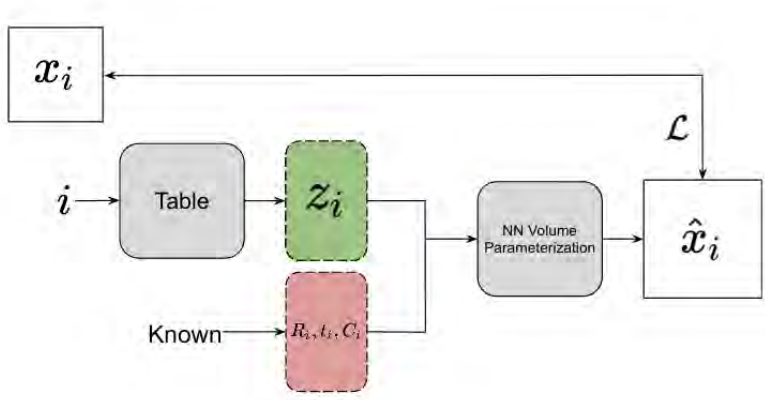


Figure 4: Experimental setup of the variational lookup table. The architecture is similar to Figure 2, with the exception that the input x_i to the network and its subsequent encoding with Enc_ξ are replaced with the use of the index i of the image as input to the latent space sampling function. The loss function $\mathcal{L}$ compares the image x_i and the generated output of the network $\hat{x}_i$.

The result of the VLT experiment, using a normal Gaussian initialization and a learning rate of 0.01, are presented in Figure 5. Additional results with a different learning rates, as well as results where the VLT latent variables are fixed to those produced by the original CryoDRGN encoder and are presented in Appendix C. In the absence of established quantitative methods for comparison of heterogeneous structure analysis in cryo-EM[14], we resort to a qualitative examination. Qualitatively, the latent space structure and the volumes/ conformations recovered by the VLT procedure agree with the original CryoDRGN results shown in Figure 3. The classification of individual particle images to specific conformations does not always agree with that produce in the benchmark CryoDRGN run, but we note that this classification is not entirely consistent across CryoDRGN runs with different seeds and otherwise similar parameters.

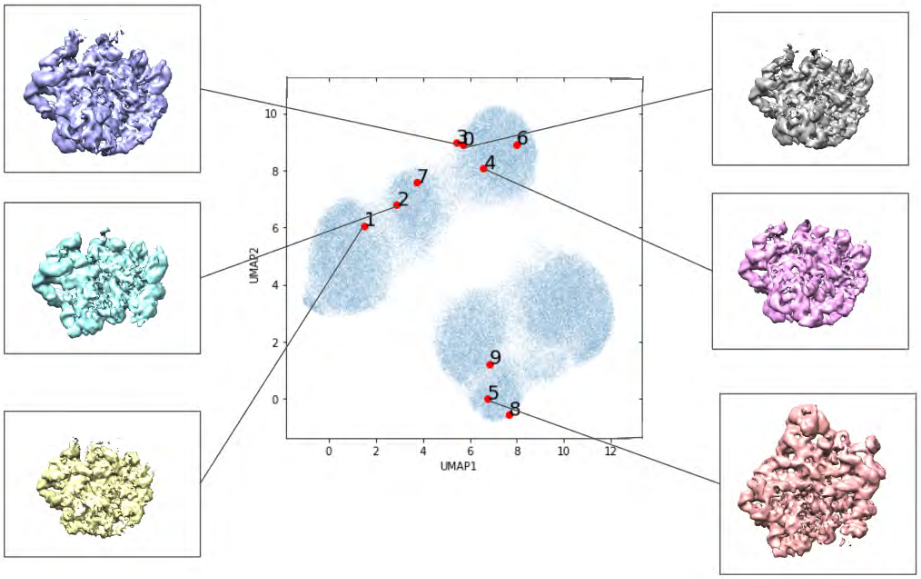


Figure 5: Random initialization of the latent space using the Variational Lookup Table architecture with latent space optimization. Labeled locations in the learned latent space of the first ten images in the data set visualized with UMAP dimensionality reduction. Volumes associated with certain areas of the latent space are included.

3.3 Experimental System: "Evil Twin"

To examine the relationship of the input to the latent variables via the encoder function, we employed a series of experiments we termed "Evil Twin," aiming to qualitatively examine the ability of the VAE to overfit particle images. In these experiments, we pair each observed particle image x_i with some arbitrary "twin" $\tilde{x}_i$; the first experiment is a permutation test, where the twins are some different particle image from the dataset (no two particle images can have the same twin. The pairing is not symmetric; image x_1 may have x_2 as its evil twin, but x_2 may have x_3 as its evil twin). Importantly, the assigned evil twin does not change during the optimization process. The $\tilde{x}_i$ was always used in the forward pass through the neural network in place of the non-evil image x_i, but the loss function was calculated against the image x_i. In other words, the encoder in the VAE is shown $\tilde{x}_i$, but the decoder is expected to produce x_i. A diagram of the architecture is shown in Figure 6. The result of the permutation evil twin experiment is presented in Figure 7.

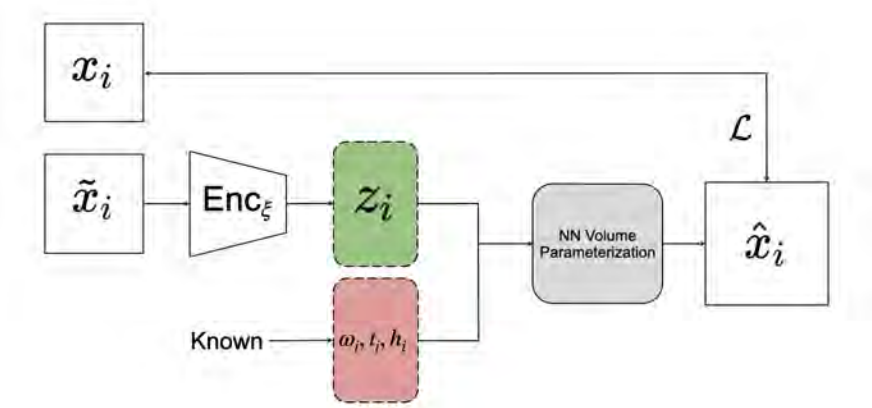


Figure 6: Experimental setup of the evil twin experiment. The architecture is similar to Figure 2, with the exception that the input to the network x_i is instead the image's chosen evil-twin $\tilde{x}_i$. The loss function $\mathcal{L}$ compares the image x_i and the generated output of the network $\hat{x}_i$.

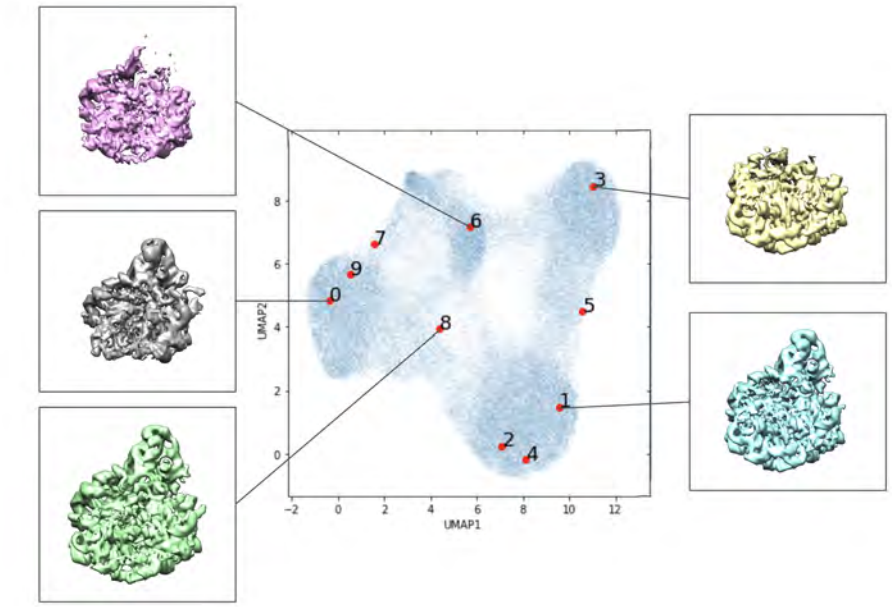


Figure 7: The evil twin experiment, permutation variant. Labeled locations in the learned latent space of the first set of images in the data set visualized in the 2-dimensional latent space. Volumes associated with certain areas of the latent space are included.

In the second evil-twin experiment we paired each particle image with a random noise image of the same dimension, where the mean and the variance of the noise were set to the emperical mean and variance of the dataset. In this experiment we used a larger encoder network to 5 encoding layers with width 1024 each, compared to the original architcture of 3 encoding layers of width 256. The results are illustrated in Figure 8.

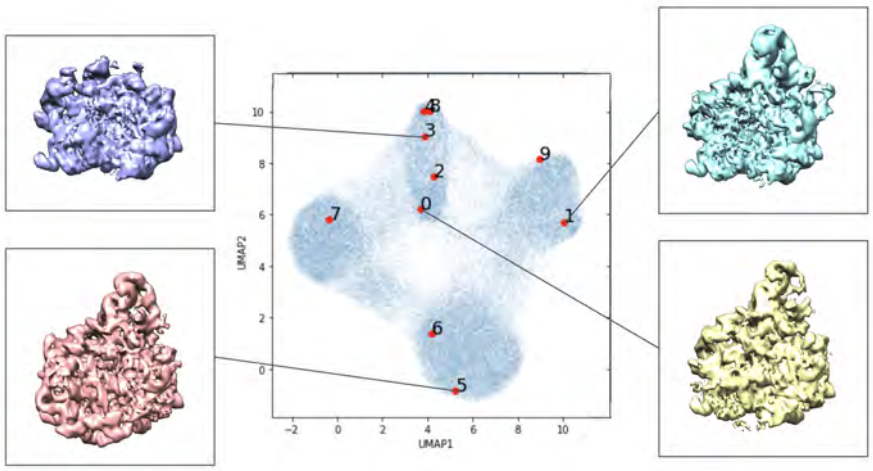


Figure 8: The evil twin experiment, noise variant. Labeled locations in the learned latent space of the first set of images in the data set visualized in the 2-dimensional latent space. Volumes associated with certain areas of the latent space are included.

We found qualitative similarity between both evil twin experiments and the baseline in Section 3.1. While the latent spaces were slightly different visually, we were able to see distinct clusters analogous to those found in the baseline and VLT experiments.

3.4 Amortization Experiments

In many applications the encoder generalizes and can be used to approximate the conditional distribution $p(z|x)$ of samples x not observed in the training phase. Evaluating this property in experimental cryo-EM data is challenging in the absence of ground-truth. Fortunately, cryo-EM provides a natural experiment: since particle freeze in arbitrary orientations, the particles in the images can show up in random orientations and are not centered. This means that an in-plane rotated particle image should be assigned the same latent variable as the original image. Due to inaccuracy in cropping of particle images from the large micrograph produced by the microscope, small in-plane translations of the particle in the particle image should not lead to large changes in the latent variable.

To test hypothesized generalization and amortization properties of the VAE, we employed a series of experiments to augment the data. In these experiments we chose to train the network in a standard manner, and then shifted all of the training images by one pixel to the right, with the rightmost column pixels rolled over to the leftmost column of the image, to serve as test images. We also performed a series of analogous experiments with rotations by using training images rotated by 90° clockwise to serve as test images. If the encoder generalizes well to images that it has not seen before, we expect it to generalize well to invariances; for example, we would expect the rotated particle images to be assigned the same latent variables as the original particle image.

We introduced the shift on the naive assumption that the edges of the images consist of only noise, and that on 128x128 images a single pixel does not represent a large enough shift that should alter the center of the image nor the image content. Similarly, a rotation of 90 degrees presents no issues of interpolation. Furthermore, because the translation and rotation of the image is fed after encoding the image to a latent space, we should expect that a translated or rotated image should result in a similar volume output from the decoder, and be mapped to a similar position in the latent space by the decoder, in keeping with the assumption of amortized learning of heterogeneous volumes.

We found that for both shifting the images by 1 pixel and rotating images by 90 degrees, these augmentations appeared to have a significant impact on where the encoder mapped images to in the latent space. There appeared to be no qualitative agreement between z_i values produced from encoding images and encoding their shifted or rotated variants. We found that shifted or rotated images still appeared to show heterogeneity and the various different structures produced were qualitatively similar across the whole latent space, but due to being mapped to very different points in the latent space, the shifted or rotated images were encoded and then decoded to create sometimes very qualitatively distinct volumes from those that the non-augmented versions produced. See Figure 9 for the shifted results, and Figure 10 for the rotated results. Some of the red labeled points appearing in the Figures also correspond to the labels in Figure 3, as the shifted and rotated results are from the same trained network as the tutorial.

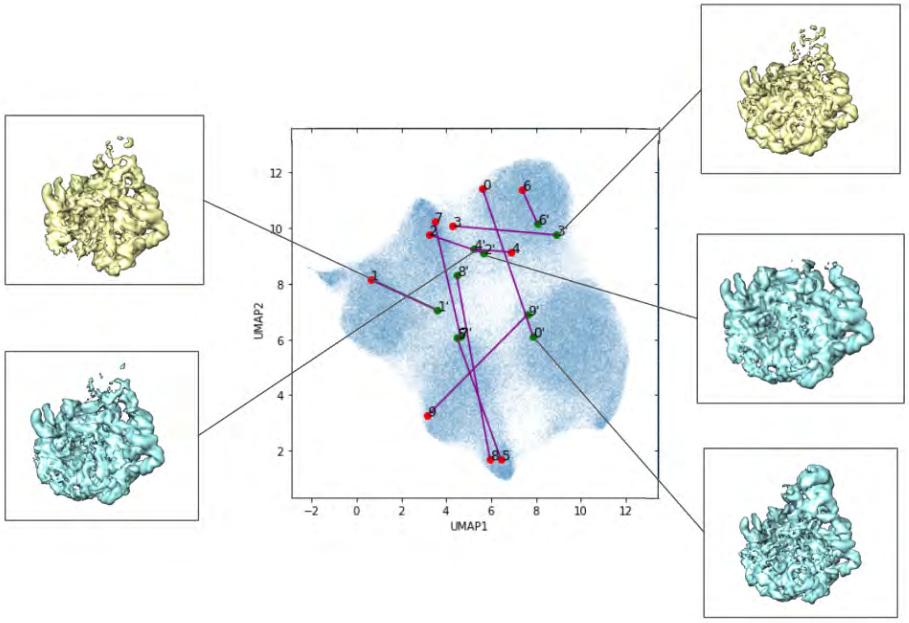


Figure 9: Shifted-by-1 images passed into a trained CryoDRGN network. Labeled locations in the learned latent space of the first set of images in the data set visualized in the 2-dimensional latent space. The first 10 training images are in red, the first 10 test images, shifted versions of the associated training images are in green. Purple lines connect the labeled images and their associated shifted test image in the latent space. Volumes associated with certain areas of the latent space are included.

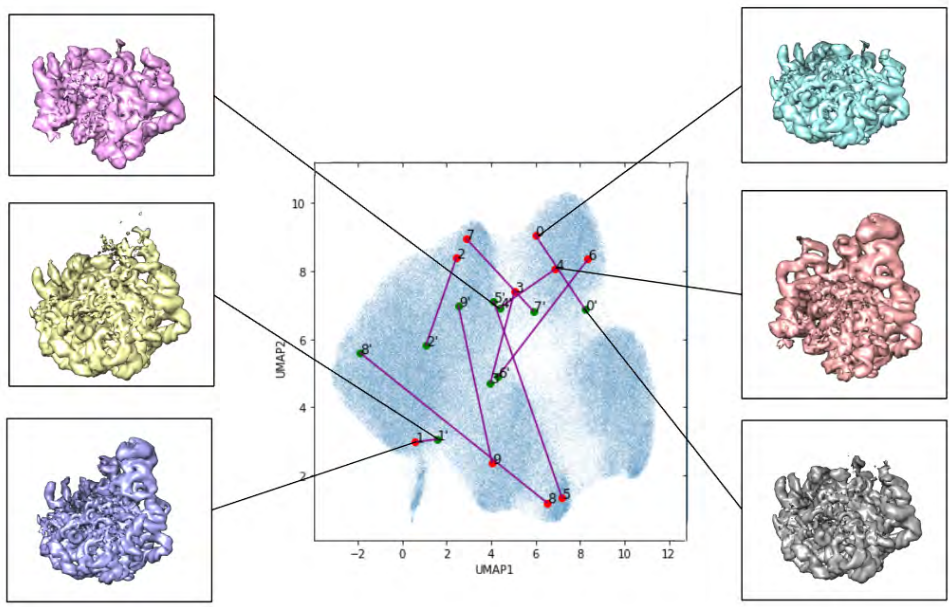


Figure 10: Rotated images passed into a trained CryoDRGN network. Labeled locations in the learned latent space of the first set of images in the data set visualized in the 2-dimensional latent space. The first 10 training images are in red, the first 10 test images, rotated versions of the associated training images are in green. Purple lines connect the labeled images and their associated rotated test image in the latent space. Volumes associated with certain areas of the latent space are included.

4 Discussion

The purpose of this case study was to examine if the common wisdom about VAEs holds is some scientific applications. We based our case study on CryoDRGN, a modified VAE developed for cryo-EM applications. We limit ourselves to a very well-studied real-world experimental setup that has been used in tutorials for this software. Where we modified the setup or the data from the baseline setup, as described above, we tried to make the minimal necessary changes, rather than optimize the architecture and parameters.

The experiments in Section 3.4 examine the ability of the encoder to generalize. In the absence of ground-truth data, we use known invariances in the cryo-EM data to examine how well the encoder generalizes to augmented data, as a proxy for generalizaton to unseen test data. The encoder does not appear to generalize well to the augmented data, suggesting that it would also not generalize well to completely new unseen test data. CryoDRGN does not enforce an invariance or approximate invariance with respect to rotations and translations in its network architecture.

In principle, it is possible to build some invariance into the network architecture, enforcing encoders that are invariant to in-plane rotations for example. This invariance has not been implemented in CryoDRGN. It is not obvious if this would be beneficial to the primary goal of recovering the different conformations, as we have not ruled out that the overfitting could be beneficial in some way in solving this problem. Furthermore, while such invariant network structures would solve the generalization problem demonstrated in our experiment (augmenting the images with rotated and translated images), it does not immediately imply that the network would generalize well to unseen particle images (truly new samples, as opposed to our augmented samples).

In the experiments in Section 3.3, we replace the input to the encoder with arbitrary images or even random noise; the original images are still used in the loss function in the comparison to the output of the encoder. Somewhat surprisingly, the algorithms still performs well (qualitatively), although this required a slightly larger network. In this case, there is no generalization that the encoder can do from one input to another

(other than some level of restriction of the latent outputs that is applicable to all the inputs). One possible interpretation is that the encoder can effectively overfit the values of the latent variable to each individual image, effectively turning the encoder into something analogous to the explicit table in Section 3.2. We note that the SNR in cryo-EM data is often less than 1/10, so noise dominates the images; we hypothesize (informally) that the encoder does not somehow "isolate" the signal in the images, but rather overfits to "noise features" as well.

In the experiment in Section 3.2, we use explicit variational approximations of latent variables (VLT) rather than an amortized encoder variational approximation. The results in the VLT setup are qualitatively similar to those in the original VAE setup. The qualitative resemblance between the outputs of the VLT and the original VAE suggests that the encoder may not be the "secret sauce" that makes the VAE produce good results. The more traditional approaches to latent variables, evaluating them explicitly (in the broad sense, including sampling from their posterior for example), may be as applicable as the encoder approach.

In Appendix B, we present analogous experiments with a classic MNIST VAE. The experiments are presented to provide some baseline in a more standard setup, and the results should be taken in this context. While there are some similarities to the case study, the results and applications are different in classic VAEs, represented by the MMIST example. The experiment in Section B.1.3 appears to indicate that explicit variables can be used in the MNIST VAE, but we did not examine in depth the generative properties of the model, and this experiment ignores the importance of the trained encoder itself as a tool that can be applied to unseen samples. MNIST images are centered and somewhat aligned, so augmentation based on translations or rotations of the MNIST images are not as good as those in CryoEM for the purpose of characterizing the generalization. In short, our conclusions have limited applicability to classic VAE applications like MMIST.

The cryo-EM problem, and especially the conformational heterogeneity aspect of cryo-EM, has several special characteristics that might make it different from some other applications; some of these may apply to other scientific applications. One characteristic that stands out is the high level of noise in particle images. Intuitively, one can argue that without a very good implicit kernel, the high level of noise make every two images very different from each other even if their clean versions were identical; in other words, one could argue that the experiment in Section 3.3 where we pair each particle image with a random noise image, is not very far from the real setup. The encoder in the cryo-EM problem also has to "deal" with very high variability in the input: particle images that should be assigned the same conformation can be taken from different viewing direction, have different in-plane rotations and translation and may be subject to different filters. Furthermore, the determination of conformations may be a poorly conditioned: it may be difficult to tell the difference between different conformations from certain viewing directions even in the absence of noise.

It should also be noted that this case study does not apply to every possible implementation of VAEs in cryo-EM applications. Indeed, CryoAI [22] appears to implement a VAE that determines the viewing direction of each particle image (but not the conformation) with good generalization properties, and CryoFire [23] may extended these results to conformational heterogeneity as well. Given the limited intended scope of this case-study, and the fact that the code for CryoAI and CryoFire was not publicly available at the time of the writing, we did not expand our investigation.

5 Conclusion

VAEs are a powerful recent tool in scientific imaging applications. Our case study experiments demonstrate that the common wisdom about the properties of VAEs and encoders might not apply directly to every application, even when the VAE architecture appears to be very successful in practice. Furthermore, our results suggest that a VAE with an amortised encoder can behave surprisingly similarly to a more traditional explicit variable, which could point to directions of future work on where each approach is preferable.

Acknowledgments

The work is supported in part by NIH grants R01GM136780, R01NS02501 and AFOSR FA9550-21-1-0317. The authors would like to thank the developers of CryoDRGN for making their code publically available.

References

[1] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," May 2014. arXiv:1312.6114 [cs, stat].

[2] D. P. Kingma and M. Welling, "An Introduction to Variational Autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019. arXiv:1906.02691 [cs, stat].

[3] B. Amos, "Tutorial on amortized optimization for learning to optimize over continuous domains," Feb. 2022. arXiv:2202.00665 [cs, math].

[4] C. Cremer, X. Li, and D. Duvenaud, "Inference Suboptimality in Variational Autoencoders," May 2018. arXiv:1801.03558 [cs, stat].

[5] M. Kim and V. Pavlovic, "Reducing the Amortization Gap in Variational Autoencoders: A Bayesian Random Function Approach," Feb. 2021. arXiv:2102.03151 [cs].

[6] C. Doersch, "Tutorial on Variational Autoencoders," Jan. 2021. arXiv:1606.05908 [cs, stat].

[7] E. D. Zhong, T. Bepler, B. Berger, and J. H. Davis, "CryoDRGN: reconstruction of heterogeneous cryo-EM structures using neural networks," *Nature Methods*, vol. 18, pp. 176–185, Feb. 2021. Number: 2 Publisher: Nature Publishing Group.

[8] V. Sandfort, K. Yan, P. M. Graffy, P. J. Pickhardt, and R. M. Summers, "Use of Variational Autoencoders with Unsupervised Learning to Detect Incorrect Organ Segmentations at CT," *Radiology: Artificial Intelligence*, vol. 3, p. e200218, July 2021. Publisher: Radiological Society of North America.

[9] N. Takeishi and A. Kalousis, "Physics-Integrated Variational Autoencoders for Robust and Interpretable Generative Modeling," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 14809–14821, Curran Associates, Inc., 2021.

[10] C. Baur, S. Denner, B. Wiestler, S. Albarqouni, and N. Navab, "Autoencoders for Unsupervised Anomaly Segmentation in Brain MR Images: A Comparative Study," Apr. 2020. arXiv:2004.03271 [cs, eess].

[11] C. Donnat, A. Levy, F. Poitevin, E. Zhong, and N. Miolane, "Deep Generative Modeling for Volume Reconstruction in Cryo-Electron Microscopy," May 2022. arXiv:2201.02867 [cs, eess, q-bio].

[12] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," Feb. 2017. arXiv:1611.03530 [cs].

[13] J.-P. Renaud, A. Chari, C. Ciferri, W.-T. Liu, H.-W. Rémigy, H. Stark, and C. Wiesmann, "Cryo-EM in drug discovery: achievements, limitations and prospects," *Nature Reviews. Drug Discovery*, vol. 17, pp. 471–492, July 2018.

[14] B. Toader, F. J. Sigworth, and R. R. Lederman, "Methods for Cryo-EM Single Particle Reconstruction of Macromolecules having Continuous Heterogeneity," Nov. 2022. arXiv:2211.10744 [q-bio].

[15] T. Bendory, A. Bartesaghi, and A. Singer, "Single-particle cryo-electron microscopy: Mathematical theory, computational challenges, and opportunities," *arXiv:1908.00574 [cs, math]*, Oct. 2019. arXiv: 1908.00574.

[16] R. R. Lederman and A. Singer, "Continuously heterogeneous hyper-objects in cryo-EM and 3-D movies of many temporal dimensions," Apr. 2017. arXiv:1704.02899 [cs].

[17] R. R. Lederman, J. Andén, and A. Singer, "Hyper-Molecules: on the Representation and Recovery of Dynamical Structures, with Application to Flexible Macro-Molecular Structures in Cryo-EM," *arXiv:1907.01589 [cs, stat]*, July 2019. arXiv: 1907.01589.

[18] E. D. Zhong, A. Lerer, J. H. Davis, and B. Berger, "CryoDRGN2: Ab Initio Neural Reconstruction of 3D Protein Structures From Real Cryo-EM Images," p. 10.

[19] E. Zhong and V. Bansal, "cryoDRGN EMPIAR-10076 tutorial," Nov. 2020.

[20] "EMPIAR-10076 CryoEM Dataset of L17-Depleted 50S Ribosomal Intermediates."

[21] A. Zadeh, Y.-C. Lim, P. P. Liang, and L.-P. Morency, "Variational Auto-Decoder: A Method for Neural Generative Modeling from Incomplete Data," Jan. 2021. arXiv:1903.00840 [cs, stat].

[22] A. Levy, F. Poitevin, J. Martel, Y. Nashed, A. Peck, N. Miolane, D. Ratner, M. Dunne, and G. Wetzstein, "CryoAI: Amortized Inference of Poses for Ab Initio Reconstruction of 3D Molecular Volumes from Real Cryo-EM Images," Aug. 2022. arXiv:2203.08138 [cs, q-bio].

[23] A. Levy, G. Wetzstein, J. Martel, F. Poitevin, and E. D. Zhong, "Amortized Inference for Heterogeneous Reconstruction in Cryo-EM," Oct. 2022. arXiv:2210.07387 [cs, q-bio].

[24] C. a. B. C. LeCun, Yann and Cortes, "MNIST handwritten digit database," *ATT Labs [Online]*, vol. 2, 2010.

Appendices

A Derivation of the ELBO

For variational autoencoders, the objective to optimize is called the evidence-based lower bound (ELBO), sometimes called the variational lower bound. We choose an inference model $q_\xi(z|x)$ with variational parameters ξ.

$$\log p_\theta(x) = \mathbb{E}_{q_\xi(z|x)}[\log p_\theta(x)] \tag{10}$$

$$= \mathbb{E}_{q_\xi(z|x)}\left[\log\left[\frac{p_\theta(x,z)}{p_\theta(z|x)}\right]\right] \tag{11}$$

$$= \mathbb{E}_{q_\xi(z|x)}\left[\log\left[\frac{p_\theta(x,z)q_\xi(z|x)}{q_\xi(z|x)p_\theta(z|x)}\right]\right] \tag{12}$$

$$= \mathbb{E}_{q_\xi(z|x)}\left[\log\left[\frac{p_\theta(x,z)}{q_\phi(z|x)}\right]\right] + \mathbb{E}_{q_\xi(z|x)}\left[\log\left[\frac{q_\xi(z|x)}{p_\theta(z|x)}\right]\right] \tag{13}$$

Where

$$D_{KL}(q_\xi(z|x)\|p_\theta(z|x)) = \mathbb{E}_{q_\xi(z|x)}\left[\log\left[\frac{q_\xi(z|x)}{p_\theta(z|x)}\right]\right] \tag{14}$$

$$\mathcal{L}_{\theta,\xi}(x) = \mathbb{E}_{q_\xi(z|x)}\left[\log\left[\frac{p_\theta(x,z)}{q_\phi(z|x)}\right]\right] \tag{15}$$

We may also write this as

$$\mathcal{L}_{\theta,\xi}(x) = \mathbb{E}_{q_\xi(z|x)}\left[\log[p_\theta(x|z)] + \log[p_\theta(z)] - \log[q_\phi(z|x)]\right] \tag{16}$$

$$= \mathbb{E}_{q_\xi(z|x)}\left[\log[p_\theta(x|z)]\right] - D_{KL}(q_\xi(z|x)\|p_\theta(z)) \tag{17}$$

B MNIST Experiments

B.1 MNIST

We performed a similar set of experiments using a small network and the MNIST dataset [24]. These experiments are presented only as a better-known baseline, they do not exhibit all the same phenomena (see discussion). We cleaned and preprocessed the dataset. We used Pytorch to construct an encoder with three hidden layers, width 256, ReLU activation, and used two separated layers at the end to create the z_μ and z_σ variables. We then passed these two variables z_μ and z_σ to a sampling function $Sample()$ to get a resultant z variable for each input in a batch:

$$\sigma = \exp(\frac{1}{2}z_\sigma) \tag{18}$$

$$\epsilon \sim \mathcal{N}(0,1) \tag{19}$$

$$z = z_\mu + \sigma \cdot \epsilon. \tag{20}$$

To create a random variable with mean z_μ and variance $\exp(z_\sigma)$. The decoder is two hidden layers, ReLU activation, with sigmoid activation for the final result.

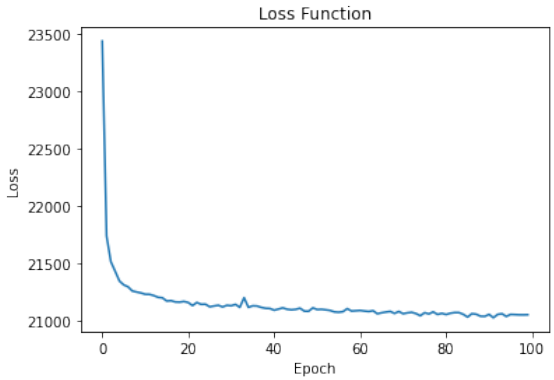
The code will be made available on our Github repo.

B.1.1 Purpose

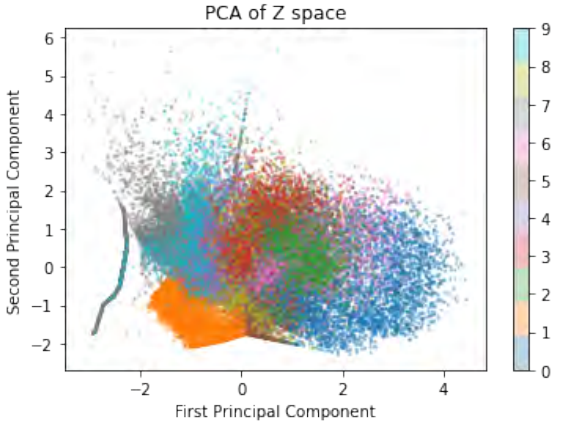
One advantage of the MNIST dataset over the CryoDRGN dataset besides being smaller and easier to work with is that we know what the resultant image "should" look like. In the case of the cryo-EM datasets, the images are often too noisy to know if we are looking at a "correct" encoding of the image, i.e. that an image from structure A is encoded into a z value that produces structure A in the trained model. In MNIST, the images are only signal, no noise, so we know if the number in the image is correctly encoded and then decoded into the appropriate number.

B.1.2 Standard VAE

The standard VAE produced results as we would expect: it got most of the images "correct" in that they are mapped by the decoder to an image that looks close enough to what the input looks like. We can see in the figures below that the model is run to approximate convergence, and the latent z space is approximately clustered by number. For numbers like 0 and 1 we saw more separation than for numbers like 6 and 8, which was expected. We also generated a z space that does not look very normally distributed, and has a few features that are very "non-normal" looking, like lines. Different runs could make the shapes look particularly sharp rather than rounded/normal shape that we expect from clusters in a normal distribution. We can also see that the network tended to show poor separation for distinguishing 4s vs 9s, 8s, 6s, and other numbers that were not as sharp as 1 or as rounded as 0, but rather had some "hybrid" structure in them. The results also appeared to be very sensitive to the random initializations, and there were a small number of cases where the results will not converge at all or will find very poor local minima they cannot escape. It was also very sensitive to learning rate, sometimes shooting off to very large values within a single batch. Loss function values can change by 20-30% between iterations regularly after convergence. See Figure 11 for these results, and Figure 12 for some example images and their reconstructions.

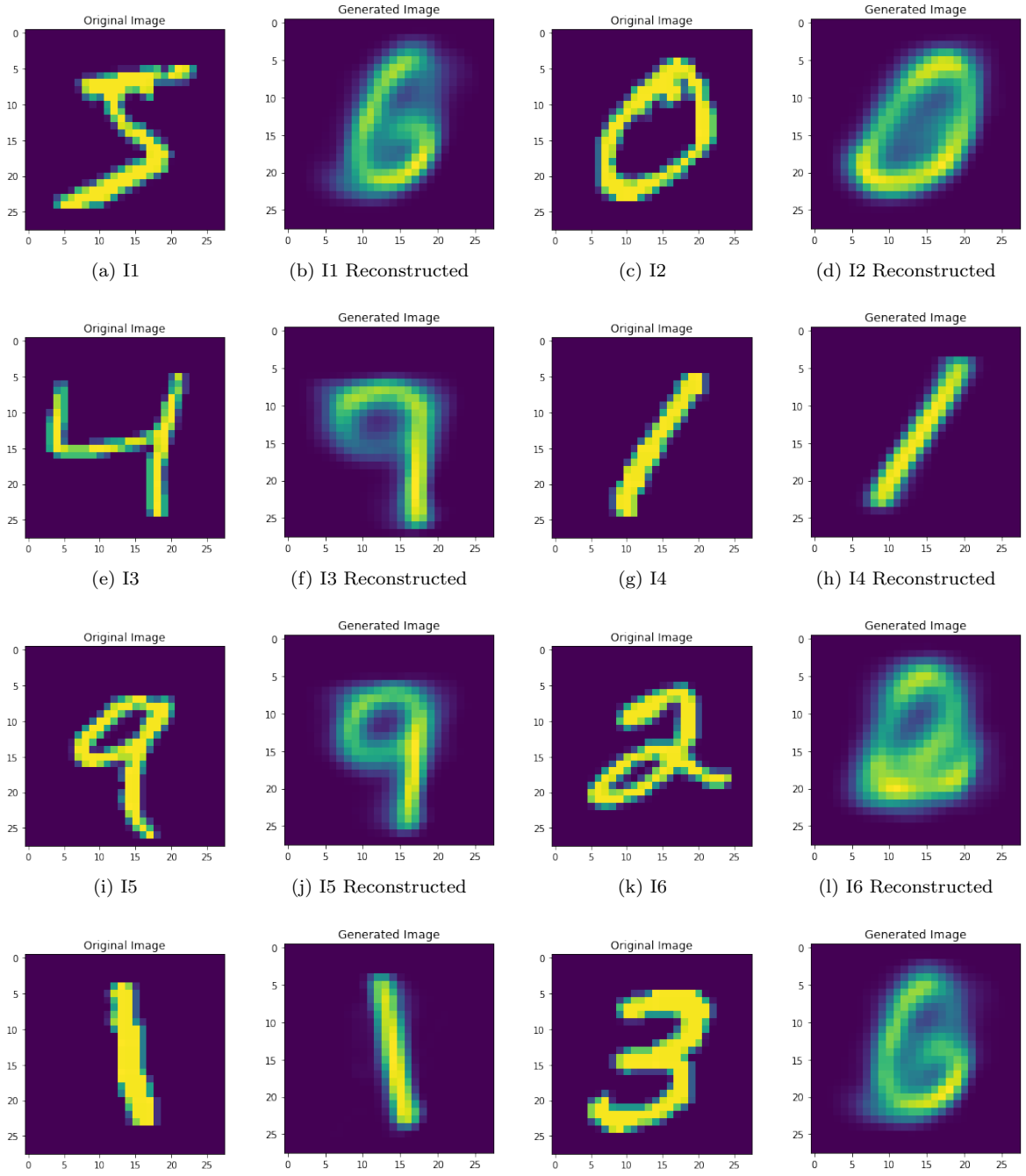


(a) MNIST VAE Loss function, run to convergence after 100 epochs



(b) The 2D latent space learning by the MNIST VAE, the Principal components are equivalent to the latent space since there is no dimensionality reduction. We can see the separation of some distinct numbers

Figure 11: The Standard MNIST VAE



(a) I1 (b) I1 Reconstructed (c) I2 (d) I2 Reconstructed

(e) I3 (f) I3 Reconstructed (g) I4 (h) I4 Reconstructed

(i) I5 (j) I5 Reconstructed (k) I6 (l) I6 Reconstructed

(m) I7 (n) I7 Reconstructed (o) I8 (p) I8 Reconstructed

Figure 12: Sampled images and their reconstructed versions using the trained MNIST VAE

B.1.3 VLT

The Embedding layer is a table that maps the index of an image to a 2-dimensional z_μ vector and a 2-dimensional z_σ vector which are then combined in the sampling function. We trained with an identical set of hyper-parameters and loss function, and evaluated identically.

B.1.4 Amortization Experiments

For experiments involving testing amortization such as noise and shifts, we trained the network on the standard dataset and then applied a transformation to the data to make it the new test dataset, which we ran after training. For shifted, we chose some number s for shift size, either 1 or 5 in our experiments, and applied a roll operation to the right where the last/right-most column of the image was rolled over to the left-most column. For noisy images, a similar procedure was applied by adding random Gaussian noise with a fixed σ^2 variance to every pixel of the test images. We did not use the test dataset of the torchvision distribution of MNIST, but rather the training set to ensure an easy way to compare across only the variances we made on data the network has already seen. Additionally, some experiments we chose to change the hyper-parameters with a larger network by increasing the width of the hidden layers or the number of layers, the dimensions of the latent space. Additionally, we changed the initialization of the lookup table. Finally, we chose to train the VLT by forcing the values in the embedding layer to remain constant while allowing the rest of the network (the decoder section of a standard VAE) to train with a standard optimizer and learning rate.

See Figures 13, 14 for results of the shifted-by-1 experiments. See Figures 15, 16 for results of the shifted-by-5 experiments.

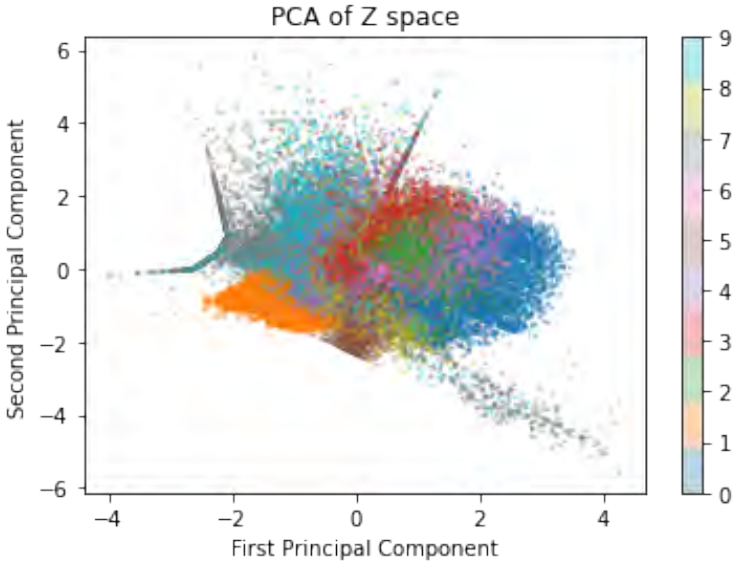
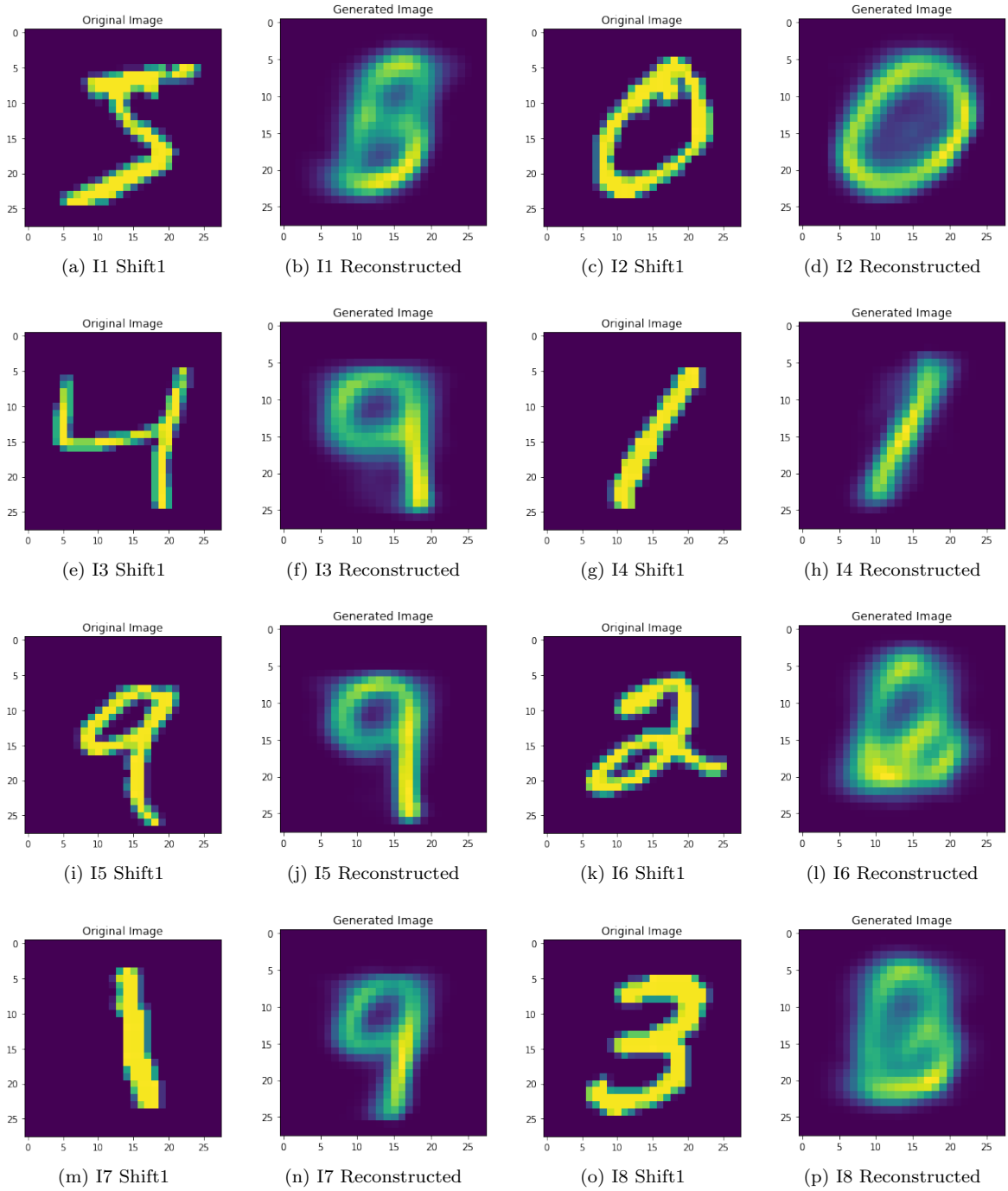


Figure 13: Shift1: The latent space of test images of the MNIST VAE, test images are original images shifted by 1 pixel to the right, with the right most pixel rolled over to the left most column.



(a) I1 Shift1 (b) I1 Reconstructed (c) I2 Shift1 (d) I2 Reconstructed

(e) I3 Shift1 (f) I3 Reconstructed (g) I4 Shift1 (h) I4 Reconstructed

(i) I5 Shift1 (j) I5 Reconstructed (k) I6 Shift1 (l) I6 Reconstructed

(m) I7 Shift1 (n) I7 Reconstructed (o) I8 Shift1 (p) I8 Reconstructed

Figure 14: Images shifted by 1 pixel and the reconstructions using the trained MNIST VAE

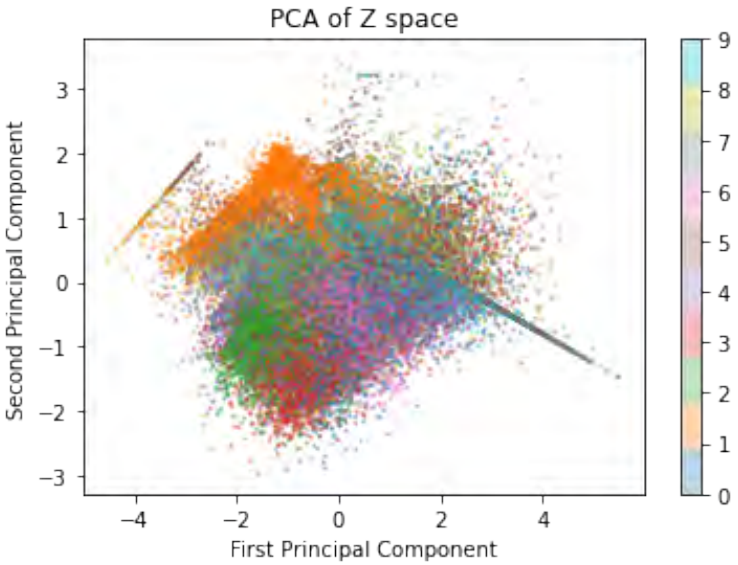
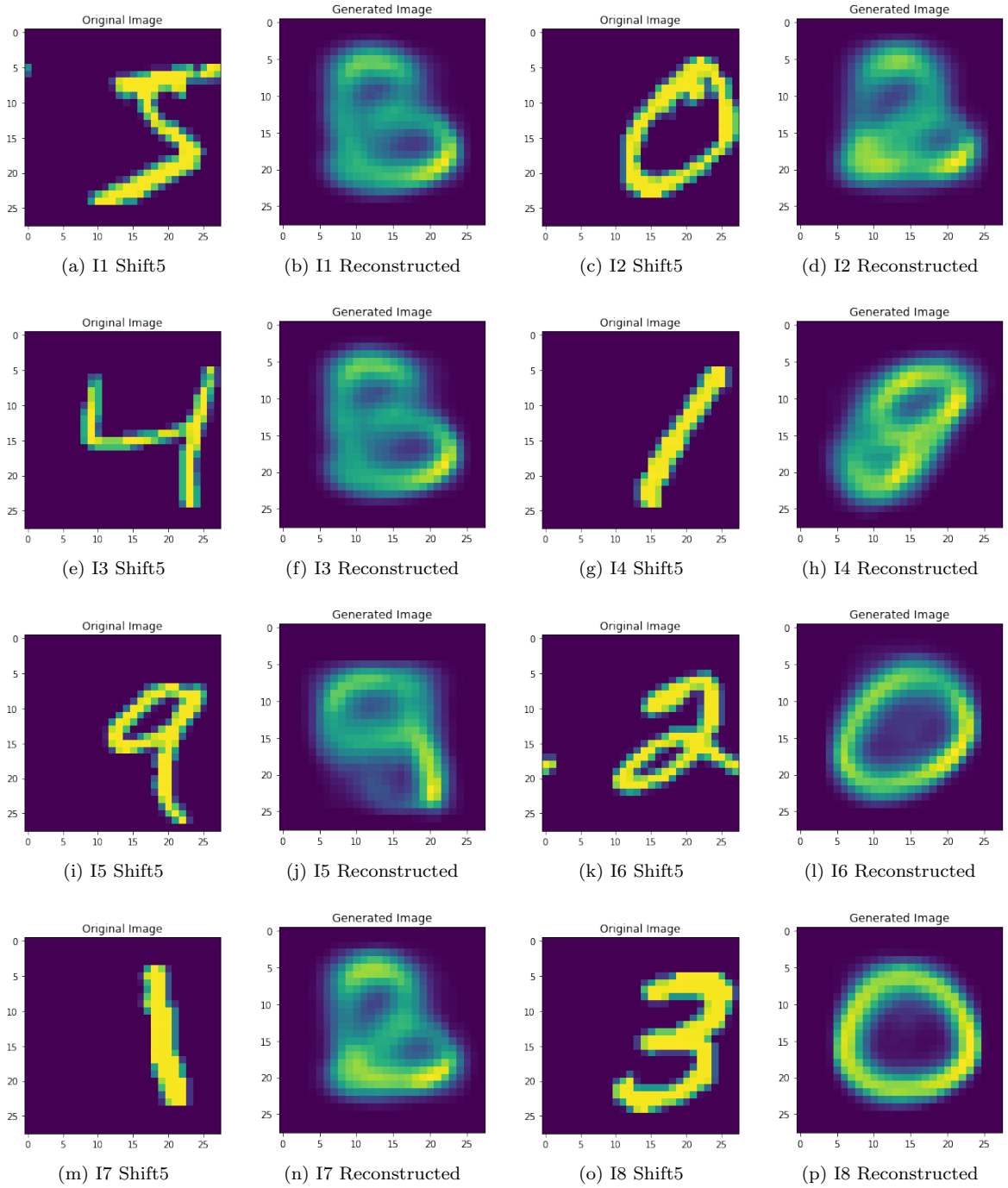


Figure 15: Shift5: The latent space of test images of the MNIST VAE, test images are original images shifted by 5 pixels to the right, with the 5 right-most columns rolled over to the left most columns.



(a) I1 Shift5 (b) I1 Reconstructed (c) I2 Shift5 (d) I2 Reconstructed

(e) I3 Shift5 (f) I3 Reconstructed (g) I4 Shift5 (h) I4 Reconstructed

(i) I5 Shift5 (j) I5 Reconstructed (k) I6 Shift5 (l) I6 Reconstructed

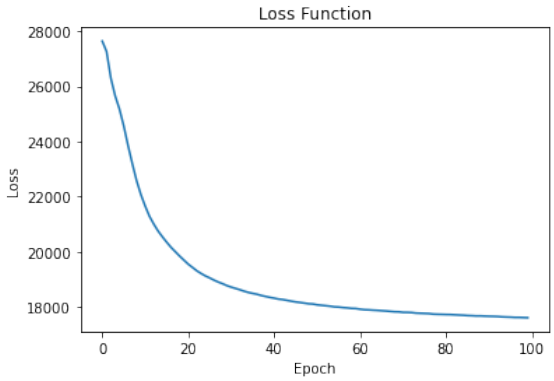
(m) I7 Shift5 (n) I7 Reconstructed (o) I8 Shift5 (p) I8 Reconstructed

Figure 16: Images shifted by 5 pixels and the reconstructions using the trained MNIST VAE

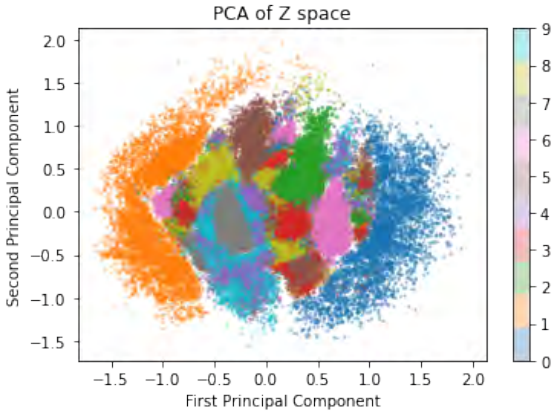
B.1.5 VLT

The VLT, Variational Lookup Table, notably took longer to converge at the same learning rate. We initialized the z to zero, so all the z_i started at the same point. Our latent z space looked significantly more normal in distribution, and we can see some more prominent clusters from 0 and 1 like in the VAE, although we also saw some more separation among the more hybrid numbers, such as 2, but in general many of the numbers did not seem to be very separated in latent space. It was not feasible to look at all the images to qualify

results, but the loss function did converge to a much lower value, since the generative loss function is much lower. The KLD loss function was higher ≈ 950 vs. ≈ 660 for the VAE model. We seemed to get fewer mistakes from the VLT, although there were some issues with confusing numbers as before, albeit less in a random sample. See Figure 17 for the latent space, and Figure 18 for some example images and their reconstructions.

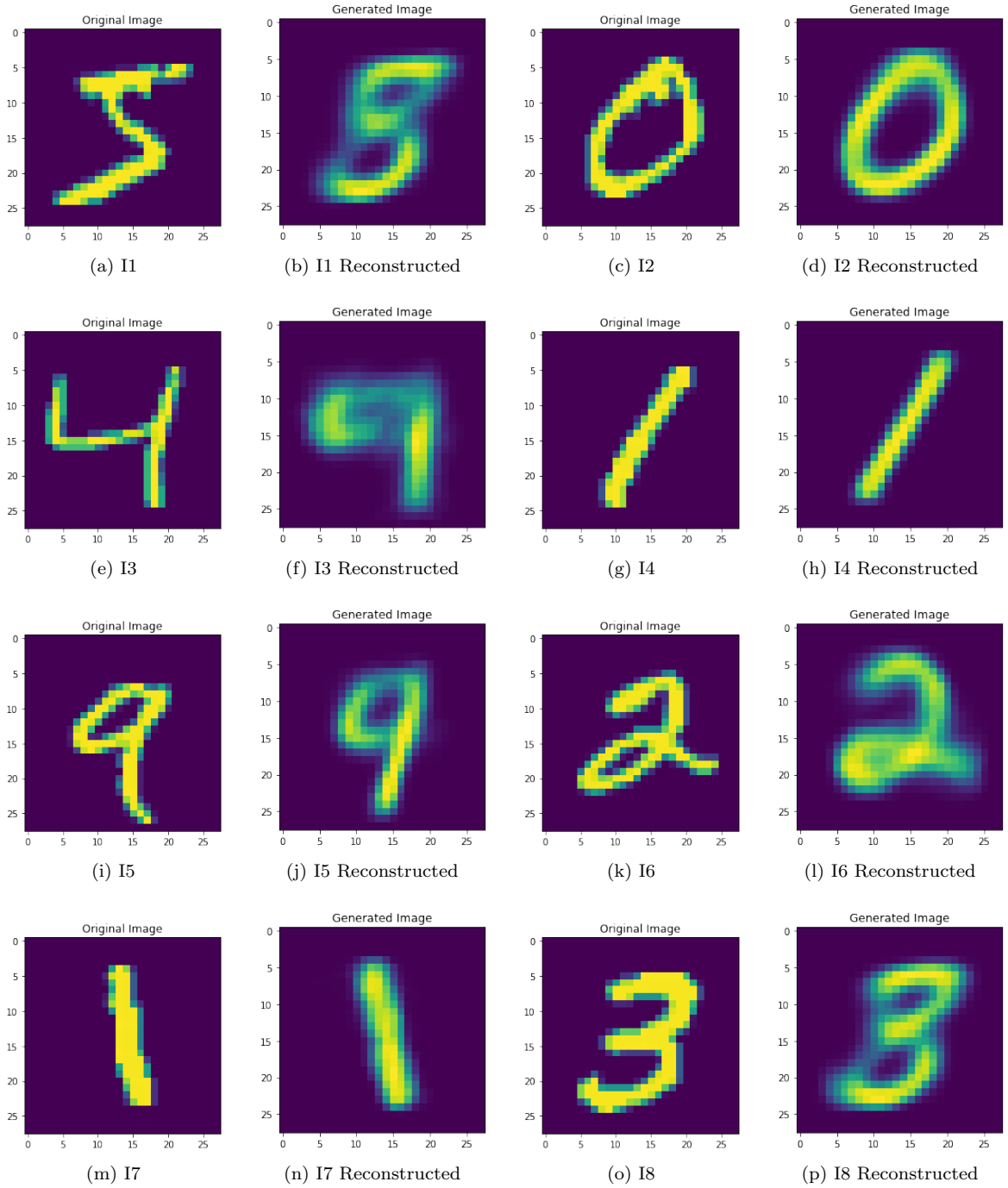


(a) MNIST VLT Loss function, run to convergence after 100 epochs



(b) 2D Latent space learned by the MNIST VLT, the latent space is in 2D so the principal components are equivalent to the dimensions. Colored by labeled digits

Figure 17: The MNIST VLT



(a) I1 (b) I1 Reconstructed (c) I2 (d) I2 Reconstructed

(e) I3 (f) I3 Reconstructed (g) I4 (h) I4 Reconstructed

(i) I5 (j) I5 Reconstructed (k) I6 (l) I6 Reconstructed

(m) I7 (n) I7 Reconstructed (o) I8 (p) I8 Reconstructed

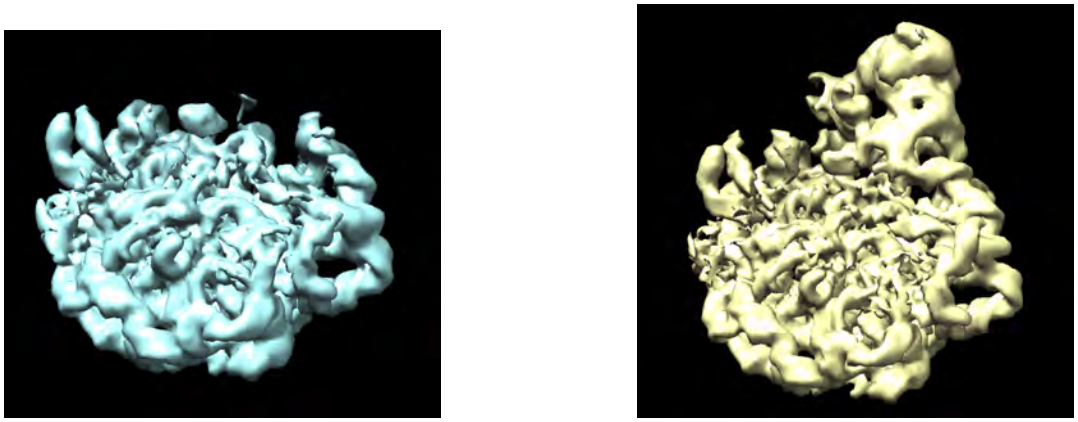
Figure 18: Sampled images and their reconstructed versions using the trained MNIST VLT

C cryo-EM Methods

All the experiments present used the EMPIAR-10076: L17-Depleted 50S Ribosomal Intermediates. This dataset was ∼130,000 images, often downsampled to 120x120 images, with about ∼Å/pixel resolution. For

the base case, we followed the CryoDRGN tutorial line-by-line. We will focus on two different types of outputs: the "encoding" of the VAE, the latent z space, and the subsequent encoding in UMAP/PCA space, since z is often in 8-10 dimensions so it becomes more interpretable when moving to dimensionality reduction. It should be noted UMAP is a random algorithm, so results can vary depending on changing seeds. The other output we will look at is the output volumes themselves, often drawn from particular regions of UMAP space.

Here we will show the plots of some of these tutorial volumes. Figure 19 shows two example volumes from the the tutorial exercise for CryoDRGN.



(a) Volume 1 (b) Volume 2

Figure 19: Volumes generated from using images 1 and 2 as generating images for volumes produced from the basic CryoDRGN training procedure.

C.1 Variational Lookup Table

The VLT experiments were performed using three distinct initializations for the table values and two learning methods, showing qualitative similarity to the standard CryoDRGN VAE. The initializations were chosen as either copied from the tutorial's trained values, random Gaussian noise, or all zeros. The two learning methods were a learning rate of 0.01 or 0. Figure 5 shows results from a random initialization with a positive learning rate. We show the other results in Appendix B. We found that when given the opportunity to optimize over the lookup table, we obtained results qualitatively similar to those from the standard tutorial run, in both apparent heterogeneity of reconstructed volumes as well as similar looking volumes in shape and some more prominent details. Quantitative assessment remains a challenge due to the nature of the unsupervised problem, but qualitative observation also showed a similar latent space visualization when mapped to 2D with UMAP dimensionality reduction technique, as well as similar relative locations of some labeled images within the lower-dimension projected latent space.

There were three types of initializations, two types of training. Pretrained initialization means that z was initialized with the same values as the resultant z values of the tutorial run, which did produce heterogeneous volumes. Random means that the z was initialized from a random normal, where the mean and variance were the mean and variance of the pretrained dataset. Zero means that all z_μ values were set to 0, and z_σ were set to -8 (we chose this value because this corresponded to a very small variance, typical values for this problem have $z_\sigma \approx -5$. Setting these to 0 created a significant issue with training since we would end up with a large variance. This initialization was also amenable, but the zero initialization was equivalent to pulling values from an $N(0, c)$ distribution where c was much smaller than 1. Everything was trained just as in the tutorial set, with the same $\approx$ 130k images, the default decoder learning rate, batch size, although training is performed over 100 epochs instead of 50. For each initialization, we chose a learning rate either of 0.01 or 0, i.e. no optimization was performed and the results are static, only the decoder was trained. We then evaluated our results in three ways: the nature of the reconstructions achieved (i.e. whether there

were heterogeneous structures present or only a homogeneous consensus reconstruction was obtained), how good these volumes looked (this is much more qualitative, although there is ample room for quantitative measurements in the future), and the nature of the z space that resulted, mostly whether we saw clustering in PCA or UMAP projections of the z_μ variables.

C.1.1 Optimizing Embeddings

We start with the embeddings that are trained. In each initialization type, we obtained qualitative heterogeneous volume reconstructions results after 100 epochs of training. Figure 20 shows the results of zero-type initialization with optimization.

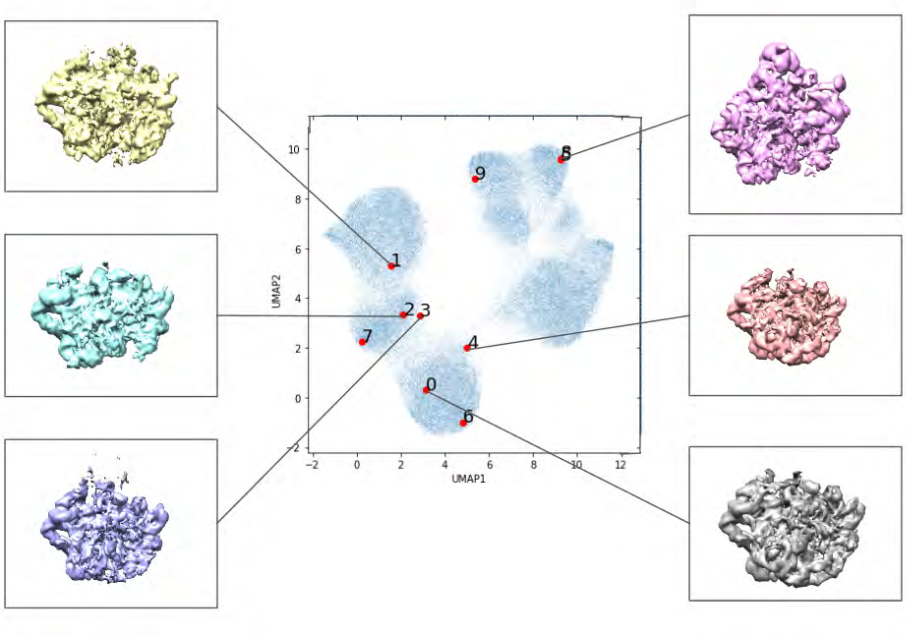


Figure 20: Zero-type initialization of the latent space using the Variational Lookup Table architecture with latent space optimization. Labeled locations in the learned latent space of the first ten images in the data set visualized with UMAP dimensionality reduction. Volumes associated with certain areas of the latent space are included.

C.1.2 No Optimization

In this method, only the pretrained initial values end up mapping to heterogeneous volumes, which was expected. The random initializations and zero initializations both mapped to a consensus homogeneous reconstruction. The reconstruction for the zero initialization is of course all the exact same, the resultant volumes for the random initialization were extremely similar, but they were not exactly the same. See Figure 21 for the results for pre-trained initialization, Figure 22 for the results for random initialization, and Figure 23 for the results for zero-type initialization with no optimization techniques for the lookup table.

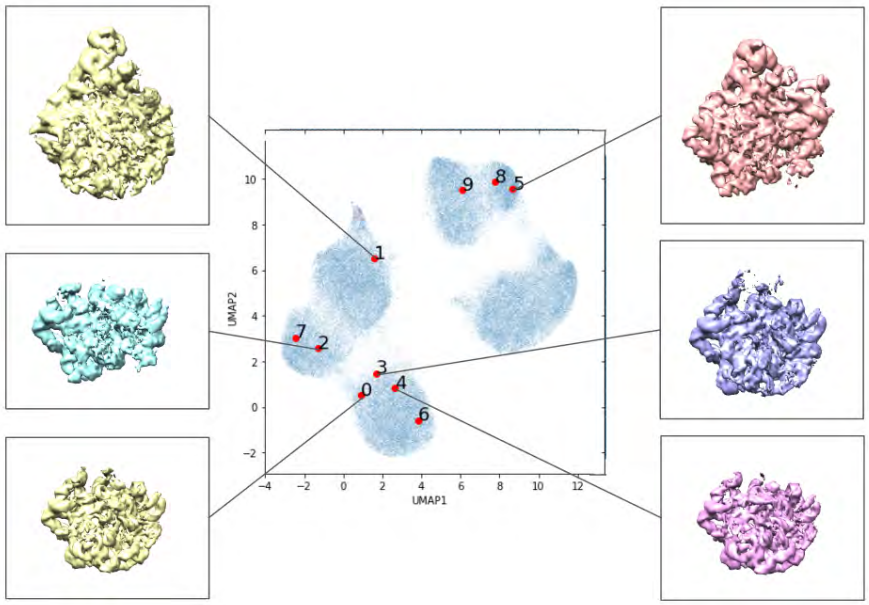
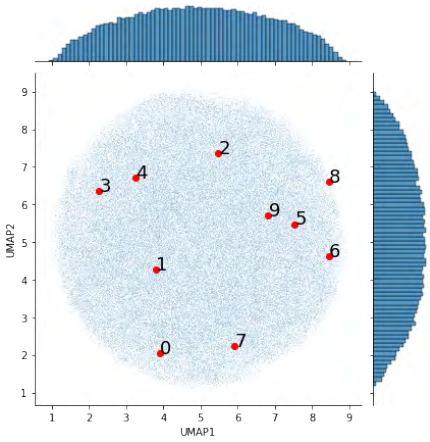
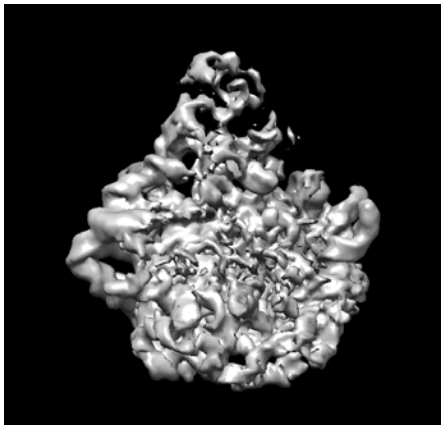


Figure 21: Pre-trained initialization of the latent space using the Variational Lookup Table architecture with no latent space optimization. Labeled locations in the learned latent space of the first ten images in the data set visualized with UMAP dimensionality reduction. Volumes associated with certain areas of the latent space are included.



(a) VLT, Random Initialization, No optimization on embedding layer, UMAP.



(b) The only unique volume produced by the VLT with random initialization, with no optimization on the embedding layer

Figure 22: The VLT with random initialization, no optimization on embedding layer

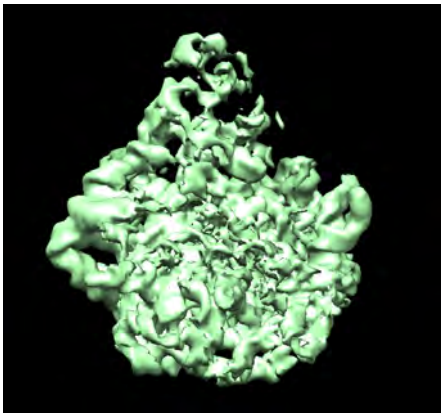


Figure 23: The only volume produced by the VLT with all points initialized to zero, with no optimization on the embedding layer

C.2 A Smaller Encoder and Latent Space

The dimensions of our z space should have naively corresponded to the degrees of freedom of our heterogeneity. For this part, we id nothing to the base tutorial except for changing the size of the encoder. In the tutorial, the encoder was fully connected, 3 layers, 256 neurons per layer. The z dimension was $\mathbb{R}^8$. Instead, we replaced the encoder with a 2 layer, 256 width network with $z \in \mathbb{R}^2$. We include these results in Figure 24

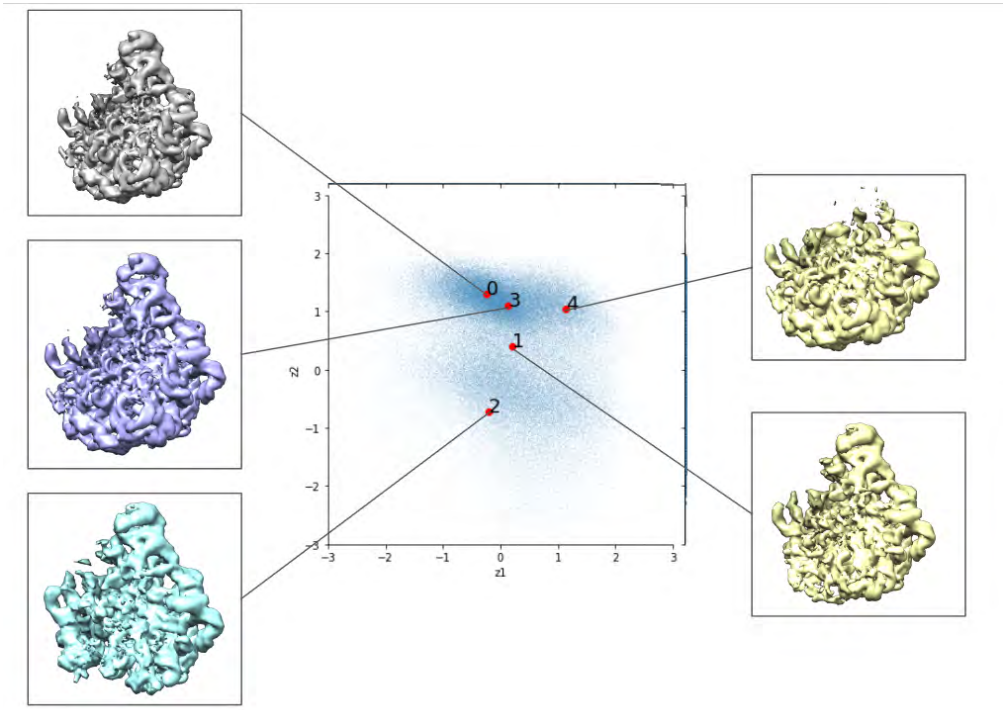


Figure 24: The standard VAE experiment with a small latent space ($d = 2$). Labeled locations in the learned latent space of the first five images in the data set visualized in the 2-dimensional latent space. Volumes associated with certain areas of the latent space are included.

It was hard to find as much heterogeneity, although it did exist, and it did not seem to be very correlated

with location or clustering in UMAP space.

C.3 Smaller Encoder, Evil Twin

We performed the same experiment as the smaller encoder and latent space experiment above, augmented with our evil twin experiment design using the perturbed training images as our evil set. The results can be seen in Figure 25.

The "Evil Twin" experiments were a series of modifications to the CryoDRGN architecture where each observed image x_i in the training set was paired with some "evil twin" image $\tilde{x}_i$. This image could be another image in the training set, although their pairings did not need to symmetric. For example, image x_1 may have x_3 as its evil twin, but x_3 may have x_4 as its evil twin, although no two images may have had the same image as its the evil twin. The evil twin may also have been randomly sampled noise. An image and its evil twin did not change throughout the training process, and were kept together through shuffling between training epochs. The $\tilde{x}_i$ was always used in the forward pass through the neural network in place of the non-evil image x_i. The loss function was calculated against the image x_i. We diagrammed this experiment in Figure 6.

For this experiment, we replace the input of the CryoDRGN model with our chosen set of images. Using the built-in minibatch method, we take the index of the images used as input and use it to sample from the original set of cryo-EM images, which we use to feed into the loss function calculation.

The Evil twin experiments were performed using two sources of evil twin images: random noise and a perturbation of the training data set, showing qualitative similarity to the standard CryoDRGN VAE. . In the case of random noise, images were created of the same dimension as the training images, with mean and variance of the noise given using the mean and variance of the dataset. For perturbed images, the training set was copied, perturbed in order, and then mapped to the original set together for training. Importantly, we found qualitative similarity with the tutorial dataset for heterogeneous reconstructions, similar to those described in the VLT experiments. While the latent spaces were slightly different in structure, we were able to see distinct clusters just as we had found both in the tutorial set and in the VLT experiments. We were also able to uncover distinct heterogeneity differences within the latent space. Both the noise and perturbation experiments showed qualitatively similar results. Figure 7 shows the results for the perturbed training set variant, Figure 8 shows the noise variant of the experiment.

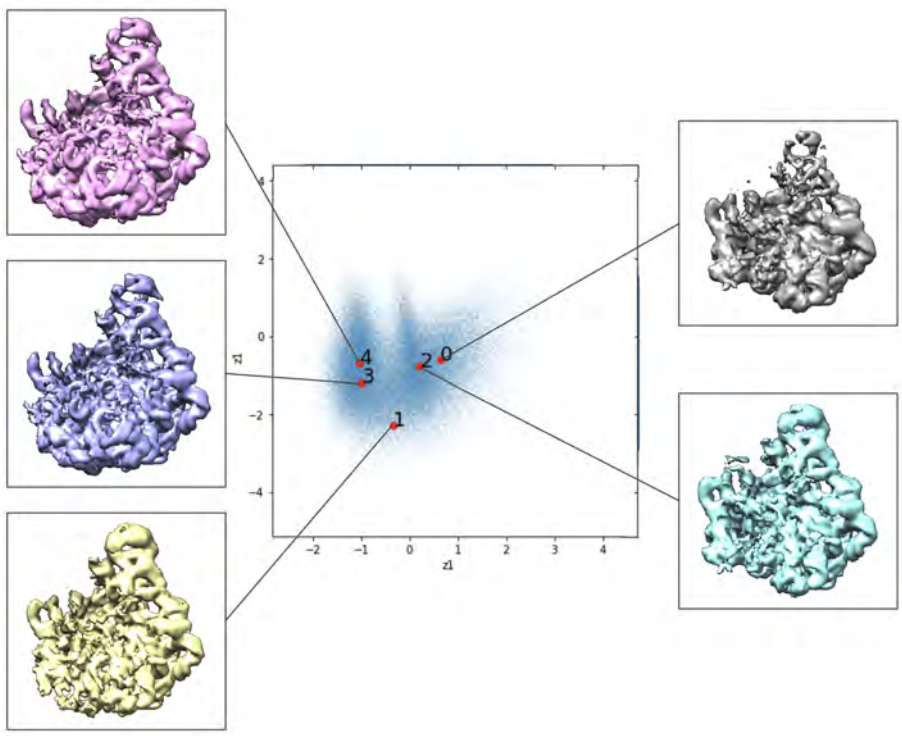


Figure 25: The evil twin experiment, perturbation variant, with a small ($d = 2$) latent space.

C.4 Positional Encoding

CryoDRGN by default uses what it calls "positional encoding" in its decoder's structure. In the forward pass, after encoding an image to some low dimensional z value, the network took this value and the rotation R_i of the image to generate a 2D lattice with $D \times D \times 3$ values, which are the $D \times D$ (x, y, z) coordinates of the lattice. Additionally, the network added another factor of D values, by taking each of these (x, y, z) pairs and applying a transformation to them, such as, for $k_j \in (x, y, z)$, $i = 0, \ldots, \frac{D}{2} - 1$:

$$pe(k_j) = \sin\left(k_j D \pi \left(\frac{2}{D}\right)^{\frac{i}{\frac{D}{2}-1}}\right) \tag{21}$$

This is one of the few different positional encodings available for training. By default, we used geom_lowf encoding of size D. We looked at the case where, all else equal, we did not supply this additional positional encoding to the decoder, and instead the network must build its images in the decoder from only the grid and the value of z in the latent space. There was little to know discernible heterogeneity among structures, as most of them have large artifacts, as an example of the second volume is shown in Figure 26.

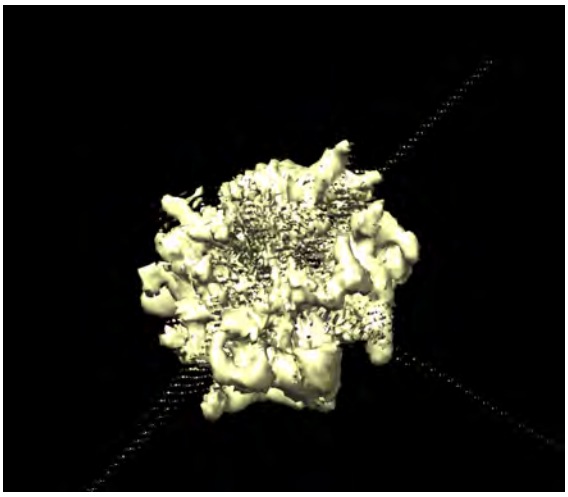


Figure 26: Volume 2 with No Positional Encoding