A Multi-Resolution Framework for U-Nets with Applications to Hierarchical VAEs

Anonymous Author(s)
Affiliation
Address
email

Abstract

U-Net architectures are ubiquitous in state-of-the-art deep learning, however their regularisation properties are understudied. In this paper, we formulate a multi-resolution framework which identifies U-Nets as finite-dimensional truncations of models on an infinite-dimensional function space. We provide theoretical results which prove that average pooling corresponds to projection within the space of square-integrable functions and show that U-Nets with average pooling implicitly learn a Haar wavelet basis representation of the data. We then leverage our framework to identify state-of-the-art hierarchical VAEs (HVAEs), which have a U-Net architecture, as forward Euler discretisations of multi-resolution diffusion sum processes which flow to a point mass, introducing instabilities. We also demonstrate that HVAEs learn a representation of time which allows for weight-sharing without detriment. We use this observation to achieve state-of-the-art HVAE performance with half the number of parameters of existing models, exploiting the properties of our continuous-time formulation.

1 Introduction

U-Net architectures are extensively utilised in modern deep learning models. First developed for image segmentation in biomedical applications [1], U-Nets have been widely applied for image-to-image translation [2], image restoration [3, 4], and multiview learning [5], amongst other tasks. They also form a core building block as the neural architecture of choice in state-of-the-art generative models, particularly for images, such as HVAEs [6, 7] and diffusion models [8, 9, 10, 11, 12, 13, 14]. In spite of their empirical success, the regularisation they impose is poorly understood.

In likelihood-based generative modelling, various model classes are competing for superiority, including normalizing flows [15, 16], autoregressive models [17, 18], diffusion models, and hierarchical variational autoencoders (HVAEs), the latter of which we focus on in this work. HVAEs form groups of latent variables with a conditional dependence structure, use a U-Net neural architecture, and are trained with the typical VAE ELBO objective (for a detailed introduction to HVAEs, see Appendix C). HVAEs show impressive synthesis results on facial images, and yield competitive likelihood performance, consistently outperforming the previously state-of-the-art autoregressive models, VAEs and flow models on computer vision benchmarks [6, 7]. HVAEs have undergone a journey of design iterations and architectural improvements in recent years, for example the introduction a deterministic backbone [19, 20, 21] and ResNet elements [22, 23] with shared parameters between the inference and generative model parts. There has also been a massive increase in the number of latent variables and overall stochastic depth, as well as the use of different types of residual cells in the decoder [6, 7] (see §4 for a detailed discussion). However, a theoretical understanding of these choices is lacking. For instance, it has not been shown why a residual backbone may be beneficial, or what the specific cell structure in VDVAEs [6] and NVAEs [7] correspond to, or how they could be improved.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In this paper we provide a theoretical framework for understanding U-Nets and apply it to HVAEs. Doing so allows us to explain various empirical findings, to relate HVAEs to diffusion processes and also to motivate a new type of model which demonstrates state-of-the-art performance with approximately half the number of parameters of a VDVAE [6]. More formally, our contributions are as follows: **(a)** We provide a multi-resolution framework for U-Nets. We formally define U-Nets as *multi-resolution bridges* over a partition of $L^2([0,1]^2)$. We prove that average pooling is a conjugate operation to projection in the Haar wavelet basis within $L^2([0,1]^2)$. We use this insight to show how U-Nets with average pooling implicitly learn a Haar wavelet basis representation of the data, characterising the regularisation within U-Nets. **(b)** We apply this framework to state-of-the-art HVAEs as an example, identifying their residual cell structure as a two-step forward Euler discretisation of a multi-resolution diffusion sum process. We uncover that this diffusion process flows to a point mass, which causes instabilities, for instance during sampling, and identify parameter redundancies through our continuous-time formulation. **(c)** In our experiments, we demonstrate these sampling instabilities and train HVAEs with the largest stochastic depth ever, achieving state-of-the-art performance with half the number of parameters by exploiting our theoretical insights. We explain these results by uncovering that HVAEs secretly represent time in their state and show that they use this information during training. We finally provide extensive ablation studies which, for instance, rule out other potential factors which correlate with stochastic depth, show the empirical gain of multiple resolutions, and find that Fourier features (a key performance driver in discrete-time diffusion models [14]) do not improve performance in the HVAE setting.

2 The Multi-Resolution Framework

A grayscale image with infinite resolution can be thought of as the graph[1] of a two-dimensional function over the unit square. To store these infinitely-detailed images in computers, we project them to some finite resolution, e.g. 512^2 pixels. These projections can still be thought of as the graphs of functions, but now a discrete one and only defined at a finite number of grid points. The relationship between the discretised version and its infinitely-fine counterpart depends entirely on how we construct this projection to preserve the details we wish to keep. One approach is to prioritise preserving the large-scale details of our images, so unless closely inspected the projection is indistinguishable from the original. This can be achieved with a multi-resolution projection [24] of the image. In this section, we introduce a *multi-resolution framework* for constructing neural network architectures that utilise such projections, prove what regularisation properties they impose, and show as an example how HVAEs with a U-Net [1] architecture can be interpreted in our framework. Proofs of all theorems and further mathematical details of the framework can be found in Appendix A.

2.1 Multi-Resolution Framework: Definitions and Intuition

What makes a multi-resolution projection good at prioritising large-scale details can be informally explained through the following thought experiment. Imagine we have an image, represented as the graph of a function, and its finite-dimensional projection drawn on the wall. The multi-resolution goodness-of-fit metric is to look at the wall, start walking away from it and stop when the image and its projection are indistinguishable by eye. The number of steps we took away from the wall can be considered a measure of 'how far away' the approximation is from the underlying function. The goal of the multi-resolution projection is therefore to have to take as few steps away as possible. The reader is encouraged to physically conduct this experiment with the examples provided in Appendix B. We can formalise the aforementioned intuitions by defining a *multi-resolution partition* [24] of sub-spaces we may project to:

Definition 1. [*Daubechies (1992)* [24]] Given a nested sequence of *approximation spaces* $\cdots \subset V_1 \subset V_0 \subset V_{-1} \subset \cdots$, $\{V_j\}_{j\in\mathbb{Z}}$ is a *multi-resolution partition* of the function space $L^2(\mathbb{R}^m)$ if: **(A1)** $\overline{\bigcup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R}^m)$; **(A2)** $\bigcap_{j\in\mathbb{Z}} V_j = \{0\}$; **(A3)** $f(\cdot) \in V_j \Leftrightarrow f(2^j\cdot) \in V_0$; **(A4)** $f(\cdot) \in V_0 \Leftrightarrow f(\cdot - n) \in V_0$.

In Definition 1, the index j references (the log of) how many steps we took in our thought experiment, so negative j corresponds to zooming in on the image. The original image[2] is a member of $L^2([0,1]^2)$,

[1] For a function $f(\,\cdot\,)$, its graph is the set $\bigcup_{x\in[0,1]^2}\{x, f(x)\}$.

[2] We here focus on grayscale, squared images for simplicity, but note that our framework can be seamlessly extended to colour images with a Cartesian product $L^2([0,1]^2) \times L^2([0,1]^2) \times L^2([0,1]^2)$.

the space of square-integrable functions on the unit square, and its finite projection to $2^j \times 2^j$ many pixels is a member of V_{-j}. The nesting property $V_{-j+1} \subset V_{-j}$ ensures that any image with $(2^{j-1})^2$ pixels can also be represented by $(2^j)^2$ pixels, but at a higher resolution. Assumption (**A1**) states that with infinitely many pixels, we can describe any infinitely detailed image. In contrast, (**A2**) says that with no pixels, we cannot approximate any images. Assumptions (**A3**) and (**A4**) allow us to form a basis for images in any V_j if we know the basis of V_0. One basis made by extrapolating from V_0 in this way is known as a *wavelet basis* [24]. Wavelets have proven useful for representing images, for instance in the JPEG standard [25], and are constructed to be orthonormal.

Now suppose we have a probability measure ν_∞ over infinitely fine images represented in $L^2([0,1]^2)$ and wish to represent this at a lower resolution. Similar to how we did for infinitely detailed images, we want to project the measure ν_∞ to a lower dimensional measure ν_j on the finite dimensional space V_{-j}. In extension to this, we want the ability to reverse this projection so that we may sample from the lower dimensional measure and create a generative model for ν_∞. We would like to again prioritise the presence of large-scale features of the original image within the lower dimensional samples. We do this by constructing a *multi-resolution bridge* from ν_∞ to ν_j, as defined below.

Definition 2. Let $\mathbb{X}$ be a compact subset of $\mathbb{R}^m$ and $\{V_{-j}\}_{j=0}^\infty$ be a multi-resolution partition of $L^2(\mathbb{X})$ scaled such that $L^2(\mathbb{X}) = \overline{\bigcup_{j\in\mathbb{N}_0} V_{-j}}$ and $V_0 = \{0\}$. Define $\mathbb{D}(L^2(\mathbb{X}))$ to be the space of probability measures over $L^2(\mathbb{X})$. A family of probability measures $\{\nu_t\}_{t\in[0,1]}$ on $L^2(\mathbb{X})$ is a *multi-resolution bridge* if:

(i) there exist increasing times $\mathcal{I} := \{t_j\}_{j\in\mathbb{N}_0}$ where $t_0 = 0$, $\lim_{j\to\infty} t_j = 1$, such that $s \in [t_j, t_{j+1})$ implies $\text{supp}(\nu_t) \subset V_{-j}$, i.e $\nu_t \in \mathbb{D}(V_{-j})$; and,

(ii) for $s \in (0,1)$, the mapping $s \mapsto \nu_s$ is continuous (in the weak sense, see Appendix A) if $s < t_j$.

The continuous time dependence in Definition 2 plays a movie of the measure ν_0 supported on V_0 growing to ν_∞, a measure on images with infinite resolution. At a time interval $[t_j, t_{j+1})$, the space V_{-j} which the measure is supported on is fixed. We may therefore define a finite-dimensional model transporting probability measures within V_{-j}, but at t_{j+1} the support flows over to V_{-j-1}. Given a multi-resolution partition, we may glue these finite models, each acting on a disjoint time interval, together in a unified fashion. In Theorem 1, we show this for the example of a continuous-time diffusion model up until some time $t_J = T \in (0,1)$ and in the *standard basis* given in the next sub-section, which will be useful in §2.3 when viewing HVAEs as discretisations of continuous-time multi-resolution diffusion models on functions.

Theorem 1. *Let $B_j : [t_j, t_{j+1}) \times \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j})$ be a linear operator (such as a diffusion transition kernel, see Appendix A) for $j < J$ with coefficients $\mu^{(j)}, \sigma^{(j)} : [t_j, t_{j+1}) \times V_{-j} \mapsto V_{-j}$, and define the natural extensions within V_{-J} in bold, i.e. $\boldsymbol{B}_j := B_j \oplus \boldsymbol{I}_{V_{-j}^\perp}$. Then the operator $\boldsymbol{B} : [0,T] \times \mathbb{D}(V_{-J}) \mapsto \mathbb{D}(V_{-J})$ and the coefficients $\boldsymbol{\mu}, \boldsymbol{\sigma} : [0,T] \times V_{-J} \mapsto V_{-J}$ given by*

$$\boldsymbol{B} := \sum_{j=0}^{J} \mathbb{1}_{[t_j,t_{j+1})} \cdot \boldsymbol{B}_j, \quad \boldsymbol{\mu} := \sum_{j=0}^{J} \mathbb{1}_{[t_j,t_{j+1})} \cdot \boldsymbol{\mu}^{(j)}, \quad \boldsymbol{\sigma} := \sum_{j=0}^{J} \mathbb{1}_{[t_j,t_{j+1})} \cdot \boldsymbol{\sigma}^{(j)},$$

induce a multi-resolution bridge of measures from the dynamics for $t \in [0,T]$ and on the standard basis as $dX_t = \boldsymbol{\mu}_t(X_t)dt + \boldsymbol{\sigma}_t(X_t)dW_t$ (see Appendix A for the details of this integration) for $X_t \in V_{-J}$, i.e. a (backward) multi-resolution diffusion process.

A U-Net [1] reduces the dimension via down-sampling operations towards a bottleneck. Layers of different dimensions are not considered to be acting on a common space. Theorem 1 states that in fact these different dimensional representations are not acting on independent spaces, they are all acting on different finite truncations of the same infinite-dimensional function space. While Theorem 1 only pertains to the backward (i.e. decoder) part of a U-net, a forward (i.e. encoder) part $\boldsymbol{F}$ can be straightforwardly defined as the reversal for all time $t \in [0,T]$. Here, $\boldsymbol{F}$ and $\boldsymbol{B}$ should be thought of as the forward and backward passes of an infinitely deep U-Net architecture [1]. We stay within V_{-J} as, in practice, we do not work in infinite dimensions, and so the diffusion above is truncated to some $T \in (0,1)$ such that the obtained resolution space V_{-J} is finite.

2.2 The regularisation property imposed by U-Net architectures with average pooling

Having defined U-Net architectures within our multi-resolution framework, we are now interested in the regularisation they impose. In practice, a pixel representation of images is used when training U-Nets, which we henceforth call the *standard basis* (see Appendix A). The standard basis is not convenient to derive theoretical results. It is instead preferable to use a wavelet basis natural to the multi-resolution partition imposed by a U-Net projection operation, which for average pooling is the *Haar basis* (see Appendix A). The Haar basis, like a Fourier basis, is an orthonormal basis of $L^2(\mathbb{X})$ which has desirable L^2-approximation properties. We formalise this idea in Theorem 2 which states that the dimension reduction operation of average pooling in the standard basis is a conjugate operation to co-ordinate projection within the Haar basis (details are provided in Appendix A).

Theorem 2. *Given V_{-j} as in Definition 1, let $x \in V_{-j}$ be represented in the standard basis $\mathbf{E}_j$ and Haar wavelet basis $\mathbf{\Psi}_j$. Let $\pi_j : \mathbf{E}_j \mapsto \mathbf{\Psi}_j$ be the change of basis map illustrated in Fig. 1, then we have the conjugacy $\pi_{j-1} \circ pool_{j,j-1} = proj_{V_{-j}} \circ \pi_j$.*

$$\begin{tikzcd}
(V_j, \mathbf{E}_j) \arrow[r, "\mathrm{Pool}_{j,j+1}"] & (V_{j+1}, \mathbf{E}_{j+1}) \arrow[d, "\pi_{j+1}"'] \\
(V_j, \mathbf{\Psi}_j) \arrow[u, "\pi_j^{-1}"] \arrow[r, dashed, "\mathrm{proj}_{V_{-j+1}}"'] & (V_{j+1}, \mathbf{\Psi}_{j+1})
\end{tikzcd}$$

Figure 1: The function space V_{-j} remains the same, but the basis changes under π_j.

Theorem 2 means that if we project an image from V_j to V_{j+1} in the Haar wavelet basis, we can alternatively view this as changing to the standard basis via π_j^{-1}, performing average pooling, and reverting back via π_{j+1} (see Figure 1). This is important because the Haar basis is orthonormal, allowing us to precisely quantify what information is lost when average pooling is used to reduce the dimension within a U-Net in Theorem 3.

Theorem 3. *Let $\{V_{-j}\}_{j=0}^J$ be a multi-resolution partition of V_{-J} with the orthogonal decompositions $V_{-j} = V_{-j+1} \oplus U_{-j+1}$. Let $F_{j,\phi}$, $B_{j,\theta} : \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j})$ be bounded linear operators with parameters ϕ and θ, for instance corresponding to a neural network. Define $\boldsymbol{F}_{j,\phi} : \mathbb{D}(V_{-j}) \mapsto \mathbb{D}(V_{-j+1})$ and let its restricted reversal be $\boldsymbol{B}_{j,\theta} : \mathbb{D}(V_{-j+1}) \mapsto \mathbb{D}(V_{-j})$ given through $\boldsymbol{F}_{j,\phi} := proj_{V_{-j+1}} \circ F_{j,\phi}$ and $\boldsymbol{B}_{j,\theta} := B_{j,\theta} \circ embd_{V_{-j}}$[3] with compositions $\boldsymbol{F}_{j_1|j_2,\phi} := \boldsymbol{F}_{j_1,\phi} \circ \cdots \circ \boldsymbol{F}_{j_2,\phi}$ and analogous $\boldsymbol{B}_{j_1|j_2,\theta}$. Then the sequence $\{\boldsymbol{B}_{1|j,\theta}(\boldsymbol{F}_{1|J,\phi}\nu_J)\}_{j=0}^J$ forms a discrete multi-resolution bridge between $\boldsymbol{F}_{1|J,\phi}\nu_J$ and $\boldsymbol{B}_{1|J,\theta}\boldsymbol{F}_{1|J,\phi}\nu_J$ at times $\{t_j\}_{j=1}^J$. Further, if $\mathcal{E} : \mathbb{D}(V_{-J}) \mapsto V_{-J}$ is a bounded linear operator, then*

$$\mathbb{E}_{x_{t_J} \sim \nu_J} \sum_{j=0}^{J} \left\| proj_{U_{-j+1}} X_{t_j} \right\|_2 / \left\| X_{t_j} \right\|_2 \leq \mathbb{E}_{x_{t_J} \sim \nu_J} \left\| \mathcal{E}(\boldsymbol{I} - \boldsymbol{B}_{1|J,\theta}\boldsymbol{F}_{1|J,\phi})(\delta_{x_{t_J}}) \right\|_2, \tag{1}$$

where $X_{t_j} := \mathcal{E}(\boldsymbol{F}_{j,\phi}\boldsymbol{F}_{J|j-1,\phi}\delta_{\{x_{t_J}\}})$.

Theorem 3 states that U-Nets push the latent data distribution to a finite multi-resolution basis, specifically a Haar basis when average pooling is used. To see this, we relate the right-hand side of Eq. 1 to the L^2 reconstruction loss by selecting $\mathcal{E}$ to be the expected position operator $\nu \mapsto \int x d\nu(x) \in L^2(\mathbb{X})$. The left-hand side of Eq. (1) represents what percentage of our data enters the orthogonal subspaces $\{U_{-j}\}_{j=0}^J$ which are discarded when using a U-Net architecture with average pooling. Theorem 3 thus shows that as we minimise the reconstruction error during training, we minimise the percentage of our data transported to the orthogonal sub-spaces $\{U_{-j}\}_{j=0}^J$. Consequently, the U-Net architecture implicitly trains our network to learn a wavelet basis of $L^2(\mathbb{X})$. This characterises the regularisation imposed by a U-Net.

These results also imply that U-Nets with average pooling provide a direct alternative to Fourier features [14, 26, 27, 28] which impose a Fourier basis, an alternative orthogonal basis on $L^2(\mathbb{X})$. However, unlike Fourier bases, there are in fact a multitude of wavelet bases which are all encompassed by our framework, and Theorem 3 pertains to all of them. This opens the door to exploring conjugacy operations beyond average pooling induced by other, potentially learned, wavelet bases optimised for specific data types.

2.3 Example: HVAEs are Sum Representation Diffusion Discretisations

To show what practical inferences we can derive from our multi-resolution framework, we apply it to analyse state-of-the-art HVAE architectures (see Appendix C for an introduction), identifying

[3] $\mathrm{embd}_{V_{-j}}$ is the natural embedding map, equivalent to $I_{V_{-j+1}} \oplus 0$ when acting on V_{-j+1}.

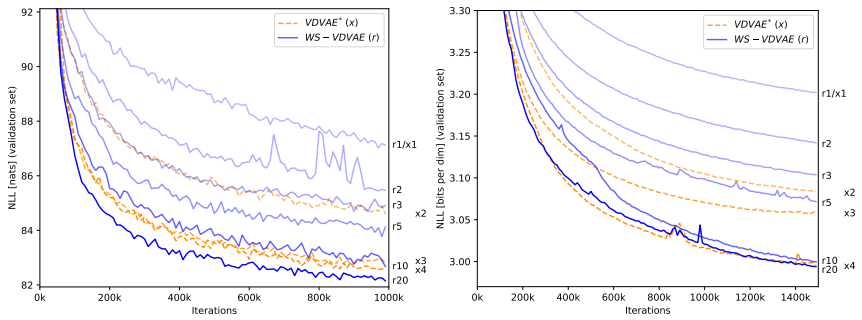

Figure 2: A small-scale study on parameter efficiency of HVAEs. We compare models with with 1,2,3 and 4 parameterised blocks per resolution ($\{x1, x2, x3, x4\}$) against models with a single parameterised block per resolution weight-shared $\{2, 3, 5, 10, 20\}$ times ($\{r2, r3, r5, r10, r20\}$). We report NLL ($\downarrow$) measured on the validation set of MNIST [left] and CIFAR10 [right]. NLL performance increases with more weight-sharing repetitions and surpasses models without weight-sharing but with more parameters.

parameter redundancies and instabilities. Here and in our experiments, we focus on VDVAEs [6]. We provide similar results for Markovian HVAEs [29, 30] and NVAEs [7] (see § 4) in Appendix A.

We start by inspecting VDVAEs. As we show next, we can tie the computations in VDVAEs to the forward and backward operators $F_{j,\phi}$ and $B_{j,\theta}$ within our framework and identify them as a two-step forward Euler discretisation of a diffusion sum process. When used with a U-Net, as in a VDVAE, this creates a *multi-resolution diffusion bridge* by Theorem 4:

Theorem 4. *Let $t_J := T \in (0, 1)$, and consider $\boldsymbol{B}_{\theta,1|J} : \mathbb{D}(V_{-J}) \mapsto \mathbb{D}(V_0)$ and its reversal $\boldsymbol{F}_{\phi,J|1} : \mathbb{D}(V_{-J}) \mapsto \mathbb{D}(V_0)$ given through the coupled multi-resolution Markov sum process in the standard basis:*

$$dX_t = \overrightarrow{\mu}_t(X_t)dt + \overrightarrow{\sigma}_t(X_t)dB_t, \qquad dZ_t = \overleftarrow{\mu}_t(Y_t)dt + \overleftarrow{\sigma}_t(Y_t)dB_t, \tag{2}$$

where $Y_0 = 0$, $dY_t = Z_t dt$, $\|Y_t\|_2 > \|Y_s\|_2$ with $0 \leq s < t \leq T$ and for a measure $\nu_J \in \mathbb{D}(V_{-J})$ we have X_T, $Z_0 \sim \boldsymbol{F}_{\phi,J|1}\nu_J$. Then, VDVAEs approximate this process, and its residual cells are a two-step forward Euler discretisation of these coupled Stochastic Differential Equations (SDEs).

To better understand Theorem 4, we visualise its residual cell structure and the corresponding discretisation steps of VDVAEs, NVAEs and Markovian HVAEs in Appendix A. Note that this process is Markov on the *sum* of the variables Z_i, not the variables themselves, which has two immediate benefits over a regular diffusion: *First*, similar sum processes have been empirically observed as efficient first-order approximates to higher-order chains, for example the memory state in LSTMs [31]. Further, VDVAEs and NVAEs are even claimed to be high-order chains (Eqs. (2,3) in [6] and Eq. (1) in [7]), despite only approximating this with a sum process. *Second*, the discretisation of the sum process has better stability under the assumption that the Y_i are increasing in magnitude. This is because the two-step splitting has a relative forward error which grows with $\|Z_{i+1}^{(\sigma)}\|_2/\|Z_i\|_2$ in the regular diffusion case, whilst in the sum process diffusion, it grows with $\|Z_{i+1}^{(\sigma)}\|_2/\|Y_i\|_2$. In the latter, the growth of the process increases the denominator and controls this error.

To show how VDVAEs impose the growth of the Y_t sum process, we prove that the forward pass of a VDVAE's U-Net enforces $Y_0 = 0$. This is done by identifying that the measure ν_0, which a VDVAE connects to the data ν_∞ via a multi-resolution bridge, is a point mass on the zero function. Consequently the backward pass must grow to reverse this, and the network learns this in a monotonic manner as we later confirm in our experiments (see §3.2).

Theorem 5. *Consider the coupled SDEs in Eq. (2), trained through the reconstruction error in Eq. C.38. Let $\tilde{\nu}_J$ denote the data measure and $\nu_0 = \delta_{\{0\}}$ be the initial multi-resolution bridge measure imposed by VDVAEs. If $q_{\phi,j}$ and $p_{\theta,j}$ are the densities of $B_{\phi,1|j}\boldsymbol{F}_{J|1}\tilde{\nu}_J$ and $B_{\theta,1|j}\nu_0$ respectively, then a VDVAE optimises the boundary condition $\min_{\theta,\phi} KL(q_{\phi,0,1}||q_{\phi,0}p_{\theta,1})$, where a double index indicates the joint distribution. Further, the relative backwards error of step i in the two-step forward Euler discretisation is $\varepsilon_{i+1} = \|Z_{i+1}^{(\sigma)}\|_2/\|Y_i\|_2$.*

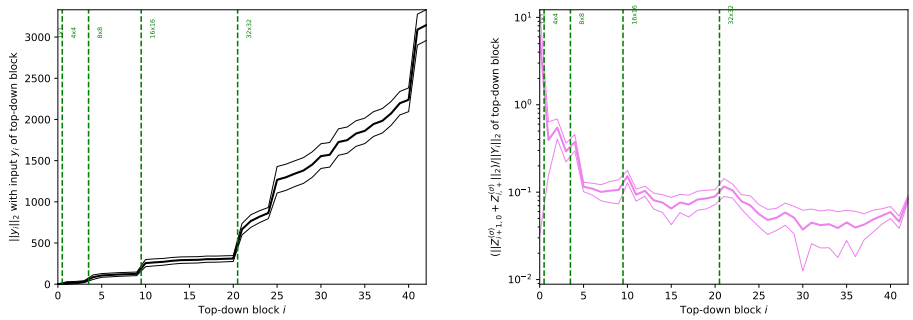

Figure 3: [Left] HVAEs secretly represent a notion of time: We measure the L_2-norm of the residual state for the backward or top-down pass (forward or bottom-up pass see Appendix I.2) over 10 batches with 100 data points each. [Right] Likewise, we measure the backward error ε_{i+1} in Theorem 5. In both plots, the thick, central line refers to the average and the thin, outer lines refer to ± 2 standard deviations.

Theorem 5 states that the VDVAE architecture forms multi-resolution bridge with the dynamics of Eq. (2), and connects our data distribution to the trivial measure on V_0: a Dirac mass at 0. In the following, we now gain two insights into this HVAE cell on instabilities and on parameter redundancies. There are two major instabilities in this discretisation. First, the imposed ν_0 is disastrously unstable as it enforces a data set, with potentially complicated topology to completely collapse at $t = 0$, and we observe the consequential sampling instability in our experiments in §3.3. Second, the discretisation error ε_i at each step of $\boldsymbol{B}_\theta$ is not explicitly imposed in VDVAEs or NVAEs, but a band-aid solution is to control this by ensuring $Y_i \xrightarrow{t \to 1} \infty$ sufficiently fast. The VDVAE architecture does, however, bolster this rate through the $Z^{(\sigma)}_{i,+}$ term, which is absent in NVAEs [7], in the discretisation steps of the residual cell (see Appendix A). We empirically observe this controlled backward error in §3.2.

Moreover, the current form of VDVAEs is over-parameterised and not informed by this continuous-time formulation. The continuous time analogue of VDVAEs [6] in Theorem 4 has time dependent coefficients $\overrightarrow{\mu}_t, \overrightarrow{\sigma}_t, \overleftarrow{\mu}_t, \overleftarrow{\sigma}_t$. We hypothesise that the increasing sum process Y_t implicitly encodes time. Hence, explicitly representing this in the model, for instance via ResNet blocks with independent parameterisations at every time step, is redundant, and a time-homogeneous model (see Appendix A for a precise formulation)— practically speaking, performing weight-sharing across time steps/layers— has the same expressivity, but requires far fewer parameters than the state-of-the-art VDVAE.

3 Experiments

In the following we probe the theoretical understanding of HVAEs gained through our framework, demonstrating its utility in four experimental analyses: (*a*) Improving parameter efficiency in HVAEs, (*b*) Time representation in HVAEs and how they make use of it, (*c*) Sampling instabilities in HVAEs, and (*d*) Ablation studies. We train HVAEs using VDVAE [6] as the basis architecture on five datasets: MNIST [32], CIFAR10 [33], two downsampled versions of ImageNet [34, 35], and CelebA [36], splitting each into a training, validation and test set (see Appendix E for details). In general, reported numeric values refer to Negative Log-Likelihood (NLL) in nats (MNIST) or bits per dim (all other datasets) on the test set at model convergence, if not stated otherwise. We note that performance on the validation and test set is generally very similar. Appendices G, H and I define the HVAE models we train, i.e. $p_\theta(\mathbf{z}_L), p_\theta(\mathbf{z}_l|\mathbf{z}_{>l}), q_\phi(\mathbf{z}_L|\mathbf{x}), q_\phi(\mathbf{z}_l|\mathbf{z}_{>l}, \mathbf{x})$ and $p_\theta(\mathbf{x}|\vec{\mathbf{z}})$, present experimental details, and additional experimental results, respectively. We provide our PyTorch code base at <see-supplementary-material> (see Appendix D for details).

3.1 "More from less": Improving parameter efficiency in HVAEs

In §2.3, we hypothesised that a time-homogeneous model has the same expressivity as a model with time-dependent coefficients, yet uses much less parameters. We start demonstrating this effect by weight-sharing ResNet blocks across time on a small scale. In Fig. 2, we train HVAEs

on MNIST and CIFAR10 with $\{1, 2, 3, 4\}$ ResNet blocks (referred to as {x1, x2, x3, x4}) in each resolution with spatial dimensions $\{32^2, 16^2, 8^2, 4^2, 1^2\}$ (VDVAE*), and compare their performance when weight-sharing a single parameterised block per resolution $\{2, 3, 5, 10, 20\}$ times (referred to as {r2,r3,r5,r10,r20}; WS-VDVAE), excluding projection and embedding blocks. As hypothesised by our framework, yet very surprising in HVAEs, NLL after 1m iterations measured on the validation set gradually increases the more often blocks are repeated even though all weight-sharing models have an identical parameter count to the $x1$ model (MNIST: 107k, CIFAR10: 8.7m). Furthermore, the weight-sharing models often outperform or reach equal NLLs compared to x2, x3, x4, all of which have more parameters (MNIST: 140k; 173k; 206k. CIFAR10: 13.0m; 17.3m; 21.6m), yet fewer activations, latent variables, and number of timesteps at which the coupled SDEs in Eq. (2) are discretised.

Table 1: A large-scale study of parameter efficiency in HVAEs. We compare our runs of VDVAE with original hyperparameters [6] (VDVAE*) against our weight-shared VDVAEs (WS-VDVAE). While WS-VDVAEs have improved parameter efficiency by a factor of 2, they reach similar NLL as VDVAE* with the simple modification inspired by our framework (weight sharing). We note that a parameter count cannot be provided for VDM [14] as the code is not public and the manuscript does not specify it.

Dataset	Method	Type	#Params	NLL ↓
MNIST 28×28	WS-VDVAE (ours)	VAE	**232k**	≤ 79.98
	VDVAE* (ours)	VAE	339k	≤ 80.14
	NVAE [7]	VAE	33m	≤ 78.01
CIFAR10 32×32	WS-VDVAE (ours)	VAE	**25m**	≤ 2.88
	WS-VDVAE (ours)	VAE	39m	≤ 2.83
	VDVAE* (ours)	VAE	39m	≤ 2.87
	NVAE [7]	VAE	131m	≤ 2.91
	VDVAE [6]	VAE	39m	≤ 2.87
	VDM [14]	Diff	–	≤ 2.65
ImageNet 32×32	WS-VDVAE (ours)	VAE	**55m**	≤ 3.68
	WS-VDVAE (ours)	VAE	85m	≤ 3.65
	VDVAE* (ours)	VAE	119m	≤ 3.67
	NVAE [7]	VAE	268m	≤ 3.92
	VDVAE [6]	VAE	119m	≤ 3.80
	VDM [14]	Diff	–	≤ 3.72
CelebA 64×64	WS-VDVAE (ours)	VAE	**75m**	≤ 2.02
	VDVAE* (ours)	VAE	125m	≤ 2.02
	NVAE [7]	VAE	153m	≤ 2.03

We now scale these findings up to large-scale hyperparameter configurations. We train VDVAEs closely following the state-of-the-art hyperparameter configurations in [6], specifically with the same number of parameterised blocks and without weight-sharing (VDVAE*), and compare them against models with weight-sharing (WS-VDVAE) and fewer parameters, i.e. fewer parameterised blocks, in Table 1. On all four datasets, the weight-shared models achieve similar NLLs with fewer parameters compared to their counterparts without weight-sharing: We use 32%, 36%, 54%, and 40% less parameters on the four datasets reported in 1, respectively. For the larger runs, weight-sharing has diminishing returns on NLL as these already have many discretisation steps. To the best of our knowledge, our models achieve a new state-of-the-art performance in terms of NLL compared to any HVAE on CIFAR10, ImageNet32 and CelebA. Furthermore, our WS-VDVAE models have stochastic depths of 57, 105, 235, 125, respectively, the highest ever trained.

3.2 HVAEs secretly represent time and make use of it

In §3.1, we showed how we can exploit insight on HVAEs through our framework to make HVAEs more parameter efficient. We now want to explain and understand this behavior further. In Fig. 3, we measure $\|Y_i\|_2$, the L_2-norm of the residual state at every bottom-up [left] and top-down [right] block with index i, over several batches for large-scale models with weight-sharing trained on MNIST (see Appendix I.2 for similar results on CIFAR10 and ImageNet32). On average, we experience an increase in the state norm across time in every resolution, interleaved by discontinuous 'jumps' at the resolution transitions (projection or embedding) where the dimension of the residual state changes. This supports our claim in §2 that HVAEs are 'sum-processes' $Y_i = \sum_{j=0} Z_j$, and hence learn to represent a notion of time in their residual state.

It is now straightforward to ask how HVAEs benefit from this sum process representation during training: As we show in Table 2, when normalising the state norm at every forward and backward

CIFAR10 ($t = 0.9$) ImageNet32 ($t = 1$) ImageNet64 ($t = 0.9$) MNIST ($t = 0.9$) CelebA ($t = 0.5$)

Figure 4: Unconditional samples (not cherry-picked) of VDVAE*. While samples on MNIST and CelebA demonstrate high fidelity and diversity, samples on CIFAR10, ImageNet32 and ImageNet64 are diverse, but are unrecognisable, demonstrating the instabilities identified by Theorem 1. Temperatures t are tuned for maximum fidelity.

block during training, i.e. forcing a "flat line" in Fig. 3.[Left], the learning deteriorates after a short while, resulting in very poor NLL results compared to the runs with a regular, non-normalised residual state above. This evidence confirms our earlier stated hypothesis that the sum process representation in ResNet-based HVAEs encodes important information which recent HVAEs heavily rely on during learning.

3.3 Sampling instabilities in HVAEs

High fidelity unconditional samples of faces, e.g. from models trained on CelebA, cover the front pages of state-of-the-art HVAE papers [6, 7]. Here, we question whether face datasets are an appropriate benchmark for HVAEs. In Theorem 5, we identified the aforementioned state-of-the-art HVAEs as flows to a point mass, hypothesising instabilities during sampling. And indeed, when sampling from our trained VDVAE* with state-of-the-art configurations, we observe high fidelity and diversity samples on MNIST and CelebA, but unrecognisable, yet diverse samples on CIFAR10, ImageNet32 and ImageNet64, in spite of state-of-the-art test set NLLs (see Fig. 4 and Appendix I.3). We argue that MNIST and CelebA, i.e. numbers and faces, have a more uni-modal nature, and are in this sense easier to learn for a discretised multi-resolution process flowing to a point mass, which is uni-modal, than the other "in-the-wild", multi-modal datasets. Trying to approximate the latter with the, in this case unsuitable, HVAE model leads to the sampling instabilities observed.

Table 2: NLL of HVAEs with and without normalisation of the residual state Y_i.

Residual state	NLL
MNIST	
Normalised (✗)	≤ 464.68
Non-normalised	≤ 81.69
CIFAR10	
Normalised (✗)	≤ 6.80
Non-normalised	≤ 2.93
ImageNet	
Normalised	≤ 6.76
Non-normalised	≤ 3.68

3.4 Ablation studies

We conducted several ablation studies which support our experimental results and further probe our multi-resolution framework for HVAEs. In this section we note key findings—a detailed account of all ablations can be found in Appendix H.4. In particular, we find that the number of latent variables, which correlates with stochastic depth, does not explain the performance observed in §3.1, supporting our claims. We show that Fourier features do not provide a performance gain in HVAEs, in contrast to state-of-the-art diffusion models, where they are essential for state-of-the-art performance [14]. This is consistent with our framework's finding that a U-Net architecture with pooling is already forced to learn a Haar wavelet basis representation of the data, hence introducing another basis does not add value. Furthermore, we demonstrate that residual cells are crucial for the performance of HVAEs as they are able to approximate the dynamics of a diffusion sum process. We also investigate synchronous vs. asynchronous processing in time between the forward and backward pass. We conclude this section by highlighting a key result which underlines the benefit of a multi-resolution bridge compared to a single-resolution model.

Table 3: Single- vs. multi-resolution HVAEs.

# Resolutions	# Params	NLL
MNIST		
Single	328k	≤ 81.40
Multiple	339k	≤ 80.14
CIFAR10		
Single	39m	≤ 2.89
Multiple	39m	≤ 2.87
ImageNet32		
Single	119m	≤ 3.68
Multiple	119m	≤ 3.67

On the effect of a multi-resolution bridge. State-of-the-art HVAEs have a U-Net architecture with pooling and, hence, are multi-resolution bridges (see Theorem 4). We investigate the effect of multiple resolutions in HVAEs (here with spatial dimensions $\{32^2, 16^2, 8^2, 4^2, 1^2\}$) against a single resolution (here with spatial dimension 32^2). We choose the number of blocks for the single resolution model such that they are distributed in the encoder and decoder proportionally to the multi-resolution model and the total number of parameters are equal in both, ensuring a fair comparison. As we show in Table 3, the multi-resolution models outperform their single-resolution counterparts, while additionally having a representational advantage due to their Haar wavelet basis representation (illustrated in Appendix I.4). This underlines the importance of the inductive bias a multi-resolution bridge imposes, as described by our framework, relative to a single-resolution model.

4 Related work

U-Nets. A U-Net [1] is an autoencoding architecture with multiple resolutions where skip connections enable information to pass between matched layers on opposite sides of the autoencoder's bottleneck. These connections also smooth out the network's loss landscape [37]. In the literature, U-Nets tend to be convolutional, and a wide range of different approaches have been used for up-sampling and down-sampling between resolutions, with many using average pooling for the down-sampling operation [8, 9, 11, 12, 14]. In this work, we focus on U-Nets as linear operators interleaved by average pooling as the down-sampling operation, and we formally characterise U-Nets as multi-resolution bridges in Theorem 3. Prior to our work, the dimensionality-reducing bottleneck structure of U-Nets was widely acknowledged as being useful, however it was unclear what regularising properties a U-Net imposed. We provided these in §2.

HVAEs. The evolution of HVAEs can be seen as a quest for a parameterisation with more expressiveness than single-latent-layer VAEs [38], while achieving stable training dynamics that avoid common issues such as posterior collapse [29, 39] or exploding gradients. *Markovian HVAEs* condition each latent variable directly on only the previous one by taking samples forward [29, 30]. Markovian HVAEs suffer from stability issues even for very small stochastic depths. *Nouveau VAEs (NVAEs)* [7] and *Very Deep VAEs (VDVAEs)* [6] combine the improvements of several earlier HVAE models (see Appendix C for details), while scaling up to larger stochastic depths. Both use ResNet-based backbones, sharing parameters between the generative and recognition parts of the model. VDVAEs are the considerably simpler approach, in particular avoiding common tricks such as a warm-up deterministic autoencoder training phase or data-specific initialisation. VDVAEs achieve a stochastic depth of up to 78, improving performance with more ResNet blocks. Worth noting is that while Markovian HVAEs and NVAEs use convolutions with appropriately chosen stride to jump between resolutions, VDVAEs use average pooling. In all HVAEs to date, a theoretical underpinning which explains architectural choices, for instance the choice of residual cell, is missing.

5 Conclusion

In this work, we introduced a multi-resolution framework for U-Nets. We provided theoretical results which uncover the regularisation property of a U-Net with average pooling as implicitly learning a Haar wavelet representation of the data. We then applied our framework to HVAEs as an example, identifying them as multi-resolution diffusion sum processes flowing to a point mass, and observed parameter redundancies and instabilities. We verified the latter theoretical insights in both small- and large-scale generative modelling experiments, and in doing so trained the deepest ever HVAEs. We explained these results by showing that HVAEs learn a representation of time and performed extensive ablation studies.

An important limitation is that the proven regularisation property of U-Nets is limited to using average pooling as the down-sampling operation. Another limitation is that we only applied our framework to HVAEs, though it is possible to apply it to other model classes. It could also be argued that the lack of exhaustive hyperparameter optimisation performed is a limitation of the work as it may be possible to obtain improved results. We demonstrate, however, that simply adding weight-sharing to the hyperparameter settings given in the original VDVAE paper [6] leads to state-of-the-art performance with improved parameter efficiency, and hence view it as a strength of our results.

References

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[3] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[4] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[5] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision – ECCV 2020*. Springer International Publishing, 2020.

[6] Rewon Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. In *International Conference on Learning Representations*, 2021.

[7] Arash Vahdat and Jan Kautz. NVAE: A deep hierarchical variational autoencoder. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[9] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[10] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

[11] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[12] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[13] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[14] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational Diffusion Models. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[15] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, 2019.

[16] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018.

[17] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[18] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in Neural Information Processing Systems*, 29, 2016.

[19] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

[20] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Learning hierarchical features from deep generative models. In *International Conference on Machine Learning*, 2017.

[21] Fabian Falck, Haoting Zhang, Matthew Willetts, George Nicholson, Christopher Yau, and Chris C Holmes. Multi-facet clustering variational autoencoders. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[22] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 2016.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992.

[25] David Taubman and Michael Marcellin. *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*, volume 642. Springer Science & Business Media, 2012.

[26] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[27] Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8684–8694, 2020.

[28] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, volume 20, 2007.

[29] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[30] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[31] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[32] Yann LeCun, Corinna Cortes, and C. J. Burges. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010.

[33] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[34] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[35] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

[36] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[37] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[38] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[39] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

[40] Naftali Tishby, Fernando C. Pereira, and William Bialek. The Information Bottleneck Method. 2000.

[41] Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. BIVA: A very deep hierarchy of latent variables for generative modeling. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[42] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

[43] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, pages 646–661. Springer, 2016.

[44] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

[45] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

[46] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We state our theoretical and experimental contributions in §1. We develop our multi-resolution framework with corresponding Theorems 1 to 5 in §2 (proofs in Appendix A). We provide our experimental results in §3.
   (b) Did you describe the limitations of your work? [Yes] See §5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix F.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We stated the assumptions of all theoretical results in §2 and refer to Appendix A for further details.
   (b) Did you include complete proofs of all theoretical results? [Yes] The complete proofs of Theorems 1 to 5, as well as of all additional experimental results are provided in Appendix A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide our code, and instructions on how to download the data and reproduce the main results in the supplementary material. In particular, we refer to the `README.md` file in the code repository for further details which follow the NeurIPS Code Completeness Checklist.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendices G and E.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Due to the significant computational cost of training extremely deep HVAEs (multiple Nvidia A100 graphic cards with 40GB of GPU memory each running for 3 weeks per run), we did not perform the multiple runs per hyperparameter setting required to provide error bars for our runs. We note that this is common practice in large-scale HVAE research (compare [6, 7]). Furthermore, in their code base, VDVAE [6], which we directly base our architecture on, reported highly stable test NLL when varying the random seed, varying only in the second decimal place in bits per dim.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We discuss our computational resources used and an estimate of the total amount of compute required to reproduce our results briefly in §3, and in detail in Appendix D.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] We discuss all existing assets used in our code base in Appendix D.

(b) Did you mention the license of the assets? [Yes] We mention the license of all assets used in Appendix D.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We provide our source code with instructions on how to reproduce our results in the supplementary material. Further details on our code are provided in Appendix D.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We address this question in Appendix E.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] We likewise address this question in Appendix E.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]