entropy

MDPI

Article

Self-Supervised Variational Auto-Encoders

Ioannis Gatopoulos [1,2] and Jakub M. Tomczak [2,*]

[1] Institute of Informatics, Universiteit van Amsterdam, Science Park 904,
1098 XH Amsterdam, The Netherlands; johngatop@gmail.com

[2] Department of Computer Science, Vrije Universiteit Amsterdam, De Boelelaan 1111,
1081 HV Amsterdam, The Netherlands

* Correspondence: j.m.tomczak@vu.nl

Abstract: Density estimation, compression, and data generation are crucial tasks in artificial intelligence. Variational Auto-Encoders (VAEs) constitute a single framework to achieve these goals. Here, we present a novel class of generative models, called *self-supervised Variational Auto-Encoder* (selfVAE), which utilizes deterministic and discrete transformations of data. This class of models allows both conditional and unconditional sampling while simplifying the objective function. First, we use a single self-supervised transformation as a latent variable, where the transformation is either downscaling or edge detection. Next, we consider a hierarchical architecture, i.e., multiple transformations, and we show its benefits compared to the VAE. The flexibility of selfVAE in data reconstruction finds a particularly interesting use case in data compression tasks, where we can trade-off memory for better data quality and vice-versa. We present the performance of our approach on three benchmark image data (Cifar10, Imagenette64, and CelebA).

Keywords: deep generative modeling; probabilistic modeling; deep learning; non-learnable transformations

check for updates

Citation: Gatopoulos, I.; Tomczak, J.M. Self-Supervised Variational Auto-Encoders. *Entropy* **2021**, *23*, 747. https://doi.org/10.3390/e23060747

Academic Editor: Eric Nalisnick

Received: 23 April 2021
Accepted: 9 June 2021
Published: 14 June 2021

1. Introduction

The framework of variational autoencoders (VAEs) provides a principled approach for learning latent-variable models. As it utilizes a meaningful low-dimensional latent space with density estimation capabilities, it forms an attractive solution for generative modeling tasks. However, its performance in terms of the test log-likelihood and quality of generated samples is often dissatisfying, and thus many modifications were proposed. In general, one can obtain a tighter lower bound and, thus, a more powerful and flexible model, by advancing over the following three components: the *encoder* [1–4], the *prior* (or *marginal* over latents) [5–9], and the *decoder* [10]. Recent studies have shown that, by employing deep hierarchical architectures and by carefully designing the building blocks of the neural networks, VAEs can successfully model high-dimensional data and reach state-of-the-art test likelihoods [11–13].

In this work, we present a novel class of VAEs, called *self-supervised Variational Auto-Encoders*, where we introduce additional variables to VAEs that result from the discrete and deterministic transformations of observed images. Since the transformations are deterministic, and they provide a specific aspect of images (e.g., contextual information through detecting edges or downscaling), we refer to them as *self-supervised representations*.

The introduction of the discrete and deterministic variables allows training deep hierarchical models efficiently by decomposing the task of learning a highly complex distribution into training smaller and conditional distributions. In this way, the model allows integrating prior knowledge about the data but still enables the synthesis of unconditional samples. Furthermore, the discrete and deterministic variables could be used to conditionally reconstruct data, which could be of great use in data compression and super-resolution tasks.

We make the following contributions: (i) We propose an extension of the VAE framework by incorporating self-supervised representations of the data. (ii) We analyze the impact of modeling natural images with different data transformations as self-supervised representations. (iii) This new type of generative model (*self-supervised Variational Auto-Encoders*), which can perform both conditional and unconditional sampling, demonstrated improved quantitative performance in terms of density estimation and the generative capabilities on image benchmarks.

2. Background

2.1. Variational Auto-Encoders

Let $\mathbf{x} \in \mathcal{X}^{\mathrm{D}}$ be a vector of observable variables, where $\mathcal{X} \subseteq \mathbb{R}$ or $\mathcal{X} \subseteq \mathbb{Z}$ and $\mathbf{z} \in \mathbb{R}^M$ denote a vector of latent variables. We are interested in a latent variable model of the following form:

$$p_{\vartheta}(\mathbf{x}, \mathbf{z}) = p_{\theta}(\mathbf{x}|\mathbf{z})\ p_{\lambda}(\mathbf{z}). \tag{1}$$

For training, we require marginalizing $\mathbf{z}$ out; however, calculating the marginal likelihood $p_{\vartheta}(\mathbf{x}) = \int p_{\vartheta}(\mathbf{x}, \mathbf{z})\mathrm{d}\mathbf{z}$ is computationally intractable for non-linear stochastic dependencies. As a potential solution, a variational family of distributions could be used for approximate inference. Then, the logarithm of the marginal likelihood could be lower-bounded and the following objective function could be derived, namely, the *evidence lower bound* (ELBO) [14]:

$$\ln p_{\vartheta}(\mathbf{x}) \geq \mathop{\mathbb{E}}_{q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\ln p_{\theta}(\mathbf{x}|\mathbf{z}) + \ln p_{\lambda}(\mathbf{z}) - \ln q_{\phi}(\mathbf{z}|\mathbf{x})\right] \tag{2}$$

Where $q_{\phi}(\mathbf{z}|\mathbf{x})$ is the variational posterior (or the *encoder*), $p_{\theta}(\mathbf{x}|\mathbf{z})$ is the conditional likelihood function (or the *decoder*), $p_{\lambda}(\mathbf{z})$ is the *prior* (or *marginal*), and ϕ, θ, and λ denote parameters. The expectation is approximated by Monte Carlo sampling while exploiting the *reparameterization trick* to obtain unbiased gradient estimators [15]. The models are parameterized by neural networks. This generative framework is known as *Variational Auto-Encoder* (VAE) [1,15].

2.2. VAEs with Bijective Priors

Even though the lower-bound suggests that the prior plays a crucial role in improving the variational bounds, usually a fixed distribution is used, e.g., a standard multivariate Gaussian. While being relatively simple and computationally cheap, the fixed prior is known to result in over-regularized models that tend to ignore most of the latent dimensions [9,16,17]. Moreover, even with powerful encoders, VAEs may still fail to match the variational posterior to a unit Gaussian prior [18].

However, it is possible to obtain a rich, multi-modal prior distribution $p(\mathbf{z})$ by using a *bijective* (or *flow-based*) model [19]. Formally, given a latent code $\mathbf{z}$, a base distribution $p_V(\mathbf{v})$ over latent variables $\mathbf{v} \in \mathbb{R}^M$, and $f : \mathbb{R}^M \to \mathbb{R}^M$, consisting of a sequence of L diffeomorphic transformations (i.e., invertible and differentiable transformations), where $f_i(\mathbf{v}_{i-1}) = \mathbf{v}_i$, $\mathbf{v}_0 = \mathbf{v}$, and $\mathbf{v}_L = \mathbf{z}$, the *change of variable* can be used sequentially to express the distribution of $\mathbf{z}$ as a function of $\mathbf{v}$ as follows:

$$\log p(\mathbf{z}) = \log p_V(\mathbf{v}) - \sum_{i=1}^{L} \log \left| \frac{\partial f_i(\mathbf{v}_{i-1})}{\partial \mathbf{v}_{i-1}} \right|, \tag{3}$$

where $\left| \frac{\partial f_i(\mathbf{v}_{i-1})}{\partial \mathbf{v}_{i-1}} \right|$ is the Jacobian-determinant of the ith transformation. Thus, using the bijective prior yields the following lower-bound:

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\log p_{\theta}(\mathbf{x}|\mathbf{z}) - \log q_{\phi}(\mathbf{z}|\mathbf{x}) + \log p_V(\mathbf{v}_0) + \sum_{i=1}^{L} \log \left| \frac{\partial f_i^{-1}(\mathbf{v}_i)}{\partial \mathbf{v}_i} \right| \right]. \tag{4}$$

In this work, we utilize Real-Valued Non-Volume Preserving (RealNVP) transformations [19] as the prior; however, any other flow-based model could be used [3,20].

In the RealNVP, invertible transformations are composed of coupling layers. A single coupling layer, for a given input $\mathbf{v}_i = [\mathbf{v}_{i,a}, \mathbf{v}_{i,b}]$, processes only $\mathbf{v}_{i,b}$ to obtain $\mathbf{v}_{i+1,b} = \exp(s(\mathbf{v}_{i,a})) \odot \mathbf{v}_{i,b} + t(\mathbf{v}_{i,a})$, where $s(\cdot)$ and $t(\cdot)$ denote *scale* and *translation* neural networks, respectively. Then, the Jacobian-determinant of a single coupling layer can be calculated analytically, namely:

$$\left| \frac{\partial f_i^{-1}(\mathbf{v}_i)}{\partial \mathbf{v}_i} \right| = \prod_{j=1}^{D-d} \exp(s(\mathbf{v}_{i,a}))_j = \exp\left(\sum_{j=1}^{D-d} s(\mathbf{x}_{i,a})_j \right). \tag{5}$$

3. Method

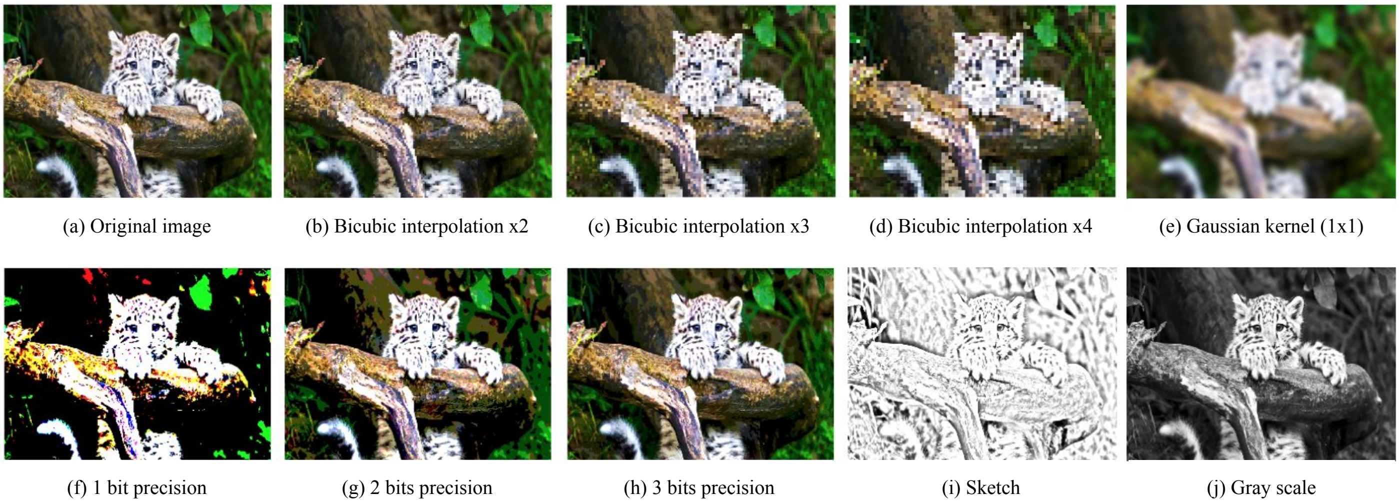
3.1. Motivation

The idea of self-supervised learning is about utilizing original unlabeled data to create additional context information. This could be achieved in multiple manners, e.g., by adding noise to data [21] or masking data during training [22]. Self-supervised learning could also be seen as turning an unsupervised model into a supervised model by, e.g., treating predicting the next pixels as a classification task [23,24]. These are only a few examples of a quickly growing research line [25].

Here, we propose to use non-trainable transformations to obtain information about the image data. Our main hypothesis is that, since working with high-quality images is challenging, we could alleviate this problem by additionally considering partial information about them. Fitting a model to images of lower quality and then enhancing them to match the target distribution appears to be an overall easier task [26,27].

By incorporating compressed transformations (i.e., the self-supervised representations) that still contain global information, with the premise that it would be easier to approximate, the process of modeling a high-dimensional complex density can be broken down into simpler tasks. In this way, the expressivity of the model will grow and gradually result in richer, better generations.

Examples of image transformations are presented in Figure 1, such as downscaling through bicubic interpolation, blurring using Gaussian kernels, considering an n-bit image representation, edge detection, or turning an RGB image into grayscale. We notice that, with these transformations, even though they discard a great deal of information, the global structure is preserved. As a result, in practice, the model should have the ability to extract a general concept of the data and to add local information afterward. In this work, we focus on downscaling (Figure 1b–d) and edge detection or *sketching* (Figure 1i).


(a) Original image (b) Bicubic interpolation x2 (c) Bicubic interpolation x3 (d) Bicubic interpolation x4 (e) Gaussian kernel (1x1)

(f) 1 bit precision (g) 2 bits precision (h) 3 bits precision (i) Sketch (j) Gray scale

Figure 1. Image Transformations. All of these transformations still preserve the global structure of the samples; however, they disregard the high resolution details in different ways.

A positive effect of the proposed framework is that the model allows us to integrate prior knowledge through the image transformations, without losing its unconditional generative functionality. Overall, we end up with a two-level VAE with three latent variables, where one is a data transformation that can be obtained in a self-supervised fashion. In Figure 2, a schematic representation of the proposed approach with downscaling is presented.

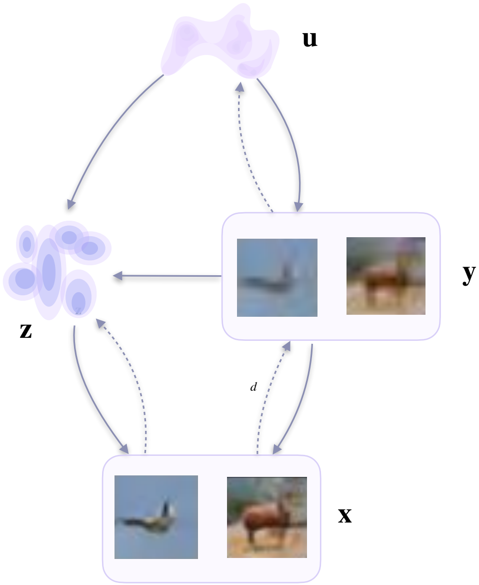


Figure 2. A schematic representation of the proposed approach.

3.2. Model Formulation

In our model, we consider representations that result from *deterministic* and *discrete* transformations of an image. Formally, we introduce a transformation $d : \mathcal{X}^D \to \mathcal{X}^C$ that takes $\mathbf{x}$ and returns an image representation $\mathbf{y}$, e.g., a downscaled image. Since we lose information about the original image, $\mathbf{z}$ could be seen as a variable that compensates for lost details in $\mathbf{x}$. Further, we propose to introduce an additional latent variable, $\mathbf{u} \in \mathbb{R}^N$, to model $\mathbf{y}$ and $\mathbf{z}$.

We can define the joint distribution of $\mathbf{x}$ and $\mathbf{y}$ as $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$, where $p(\mathbf{y}|\mathbf{x})$ becomes deterministic due to the deterministic transformation $d(\cdot)$. Formally, $p(\mathbf{y}|\mathbf{x}) = \delta(\mathbf{y} - d(\mathbf{x}))$, where $\delta(\cdot)$ denotes the Kronecker delta (for $\mathcal{X} = \mathbb{Z}$) or the Dirac delta (for $\mathcal{X} = \mathbb{R}$). Once we define the deterministic function d, for given data $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, we can obtain corresponding $\mathbf{y}$'s. At first glance, it might be not apparent what is a potential advantage of our approach except for knowing how to generate pairs of data $\{(\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^{N}$. In order to obtain insight into that, we first consider the joint entropy of $\mathbf{x}$ and $\mathbf{y}$, which is expressed as follows:

$$\begin{aligned} \mathbb{H}[\mathbf{x}, \mathbf{y}] &= \mathbb{H}[\mathbf{y}|\mathbf{x}] + \mathbb{H}[\mathbf{x}] \quad (6) \\ &= \mathbb{H}[\mathbf{x}|\mathbf{y}] + \mathbb{H}[\mathbf{y}]. \quad (7) \end{aligned}$$

Immediately, we can notice that, since $\mathbf{y}$ is a result of a deterministic function of $\mathbf{x}$, the first component in Equation (6) is 0, i.e., $\mathbb{H}[\mathbf{y}|\mathbf{x}] = 0$. This follows from the fact that $\mathbf{y}$ is a processed version of $\mathbf{x}$ and contains no new information about $\mathbf{x}$. As a result, we can express the entropy of $\mathbf{x}$ as follows:

$$\mathbb{H}[\mathbf{x}] = \mathbb{H}[\mathbf{x}|\mathbf{y}] + \mathbb{H}[\mathbf{y}]. \quad (8)$$

In other words, the entropy of the original data is equal to the conditional entropy of the original data and the entropy of the processed data. In general, modeling conditional distributions is an easier task than modeling the original data, and modeling $p(\mathbf{y})$ may be also less complicated than modeling $p(\mathbf{x})$ because $\mathbf{y}$ corresponds to a representation (a processed version) of $\mathbf{x}$. As a result, we can turn the problem of modeling a complex marginal distribution $p(\mathbf{x})$ into a self-supervised problem, where we factorize the joint distribution of $\mathbf{x}$ and $\mathbf{y}$ as $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}|\mathbf{y})p(\mathbf{y})$. Further, we propose to compensate lost information in $\mathbf{y}$ by introducing $\mathbf{z}$, and we also model $\mathbf{y}$ and $\mathbf{z}$ by introducing the additional latent variable $\mathbf{u}$. The joint distribution over all random variables in our model is as follows:

$$p(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{u}) = p(\mathbf{x}|\mathbf{y}, \mathbf{z})\ p(\mathbf{y}|\mathbf{u})\ p(\mathbf{z}|\mathbf{y}, \mathbf{u})\ p(\mathbf{u}). \tag{9}$$

All dependencies in our model are presented in Figure 2.

Eventually, for training, we need to calculate the logarithm of the marginal likelihood, namely, $p(\mathbf{x}, \mathbf{y}) = \iint p(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{u})\ d\mathbf{z}\mathbf{u}$. Calculating these integrals is intractable; therefore, we again utilize the variational inference. For this purpose, we propose to apply the following family of variational distributions over latent variables:

$$Q(\mathbf{u}, \mathbf{z}|\mathbf{x}, \mathbf{y}) = q(\mathbf{u}|\mathbf{y})q(\mathbf{z}|\mathbf{x}). \tag{10}$$

Notice that we skip a dependency between $\mathbf{u}$ and $\mathbf{x}$. Utilizing the variational inferences gives the following lower-bound on the logarithm of the marginal likelihood:

$$\begin{aligned}\ln p(\mathbf{x}, \mathbf{y}) \geq \mathbb{E}_Q \big[\ln p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z}) + \ln p(\mathbf{z}|\mathbf{u}, \mathbf{y}) + \ln p(\mathbf{y}|\mathbf{u}) + \ln p(\mathbf{u}) + \\ - \ln q(\mathbf{z}|\mathbf{x}) - \ln q(\mathbf{u}|\mathbf{y}) \big].\end{aligned} \tag{11}$$

Intuitively, the premise for self-supervised Variational Auto-Encoder (selfVAE) is that the latents $\mathbf{u}$ will capture the global structure of the input data and the latents $\mathbf{z}$ will encode the missing information between $\mathbf{y}$ and $\mathbf{x}$, guiding the model to discover the distribution of the target observations. To highlight the self-supervised part in our model, we refer to it as the *self-supervised Variational Auto-Encoder* (or selfVAE for short). Further, we propose to choose the following distributions:

$$p(\mathbf{v}) = \mathcal{N}(\mathbf{v}|\mathbf{0}, \mathbf{1}) \tag{12}$$

$$p_\lambda(\mathbf{u}) = p(\mathbf{v}) \prod_{i=1}^{F} \left| \det \frac{\partial f_i(\mathbf{v}_{i-1})}{\partial \mathbf{v}_{i-1}} \right|^{-1} \tag{13}$$

$$p_{\theta_1}(\mathbf{y}|\mathbf{u}) = \sum_{i=1}^{I} \pi_i(\mathbf{u}) \mathrm{Dlogistic}\Big(\mu_i(\mathbf{u}), s_i(\mathbf{u})\Big) \tag{14}$$

$$p_{\theta_2}(\mathbf{z}|\mathbf{y}, \mathbf{u}) = \mathcal{N}\Big(\mathbf{z}|\boldsymbol{\mu}_{\theta_2}(\mathbf{y}, \mathbf{u}), \mathrm{diag}\big(\boldsymbol{\sigma}_{\theta_2}(\mathbf{y}, \mathbf{u})\big)\Big) \tag{15}$$

$$p_{\theta_3}(\mathbf{x}|\mathbf{z}, \mathbf{y}) = \sum_{i=1}^{I} \pi_i(\mathbf{z}, \mathbf{y}) \mathrm{Dlogistic}\Big(\mu_i(\mathbf{z}, \mathbf{y}), s_i(\mathbf{z}, \mathbf{y})\Big) \tag{16}$$

$$q_{\phi_1}(\mathbf{u}|\mathbf{y}) = \mathcal{N}\Big(\mathbf{u}|\boldsymbol{\mu}_{\phi_1}(\mathbf{y}), \mathrm{diag}\big(\boldsymbol{\sigma}_{\phi_1}(\mathbf{y})\big)\Big) \tag{17}$$

$$q_{\phi_2}(\mathbf{z}|\mathbf{x}) = \mathcal{N}\Big(\mathbf{z}|\boldsymbol{\mu}_{\phi_2}(\mathbf{x}), \mathrm{diag}\big(\boldsymbol{\sigma}_{\phi_2}(\mathbf{x})\big)\Big). \tag{18}$$

For the images $\mathbf{x}$ and $\mathbf{y}$, we use Dlogistic, which is the discretized logistic distribution [28,29]. This is defined as a difference of two CDFs of the logistic distribution with a mean μ and a scale ν:

$$\mathrm{Dlogistic}(x) = \mathrm{sigm}\left(\frac{(x + 0.5 - \mu)}{\nu}\right) - \mathrm{sigm}\left(\frac{(x - 0.5 - \mu)}{\nu}\right), \tag{19}$$

where sigm is the sigmoid function, and we utilize a flow-based model for $p_\lambda(\mathbf{u})$. We use the discretized logistic distribution because the images are represented by values between

0 and 255. For integer-valued random variables, other distributions, such as Gaussian, are inappropriate.

3.3. Generation and Reconstruction in selfVAE

Latent variable generative models, such as VAEs, can be used to synthesize novel content through the following process: $\mathbf{z} \sim p(\mathbf{z}) \rightarrow \mathbf{x} \sim p(\mathbf{x}|\mathbf{z})$, i.e., we first sample $\mathbf{z}$, which is further fed to the conditional distribution to sample $\mathbf{x}$. Additionally, these models can be utilized in reconstructing a datapoint $\mathbf{x}^*$ by using the following scheme: $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}^*) \rightarrow \mathbf{x} \sim p(\mathbf{x}|\mathbf{z})$. During training, VAEs use the reconstruction process.

Interestingly, our approach allows utilizing more operations regarding data generation and reconstruction. First, analogously to VAEs, the selfVAE can generate data by applying the following hierarchical sampling process (we refer to it as *generation*, see Figure 3i): $\mathbf{u} \sim p(\mathbf{u}) \rightarrow \mathbf{y} \sim p(\mathbf{y}|\mathbf{u}) \rightarrow \mathbf{z} \sim p(\mathbf{z}|\mathbf{u},\mathbf{y}) \rightarrow \mathbf{x} \sim p(\mathbf{x}|\mathbf{y},\mathbf{z})$. In this procedure, we start with sampling $\mathbf{u}$ and proceed to sample $\mathbf{z}$ and $\mathbf{y}$ given $\mathbf{u}$, and eventually we can obtain $\mathbf{x}$. Alternatively, we can use the ground-truth $\mathbf{y}$ (i.e, $\mathbf{y}^* = d(\mathbf{x}^*)$), and sample or infer $\mathbf{z}$.

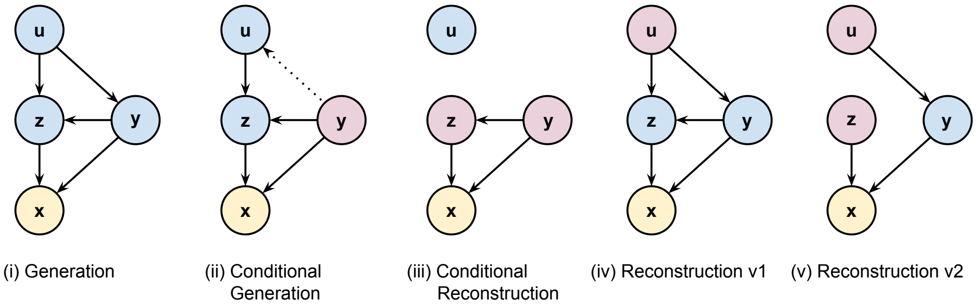


Figure 3. The generation and reconstruction schema in the self-supervised Variational Auto-Encoder. Blue and violet nodes represent the sampled and inferred latent codes, respectively, and yellow nodes correspond to the data.

Then, the generative process is the following (we refer to it as *conditional generation*, see Figure 3ii): $\mathbf{u} \sim q(\mathbf{u}|\mathbf{y}^*) \rightarrow \mathbf{z} \sim p(\mathbf{z}|\mathbf{u},\mathbf{y}^*), \rightarrow \mathbf{x} \sim p(\mathbf{x}|\mathbf{y}^*,\mathbf{z})$. Knowing $\mathbf{y}^*$ allows us to use the variational posterior to sample $\mathbf{u}$ instead of applying the prior $p(\mathbf{u})$. This scenario is possible if we consider $\mathbf{y}^*$ as a compressed version of $\mathbf{x}^*$ that is shared instead of the original content.

Similarly to the conditional generation, we can conditionally reconstruct $\mathbf{x}^*$ by utilizing $\mathbf{y}^*$. Then, the procedure is the following (we refer to it as *conditional reconstruction*, see Figure 3iii): $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}^*) \rightarrow \mathbf{x} \sim p(\mathbf{x}|\mathbf{y}^*,\mathbf{z})$. Here, we skip inferring $\mathbf{u}$, because for the given $\mathbf{y}^*$ and $\mathbf{x}^*$, the posterior could be used to sample $\mathbf{z}$ and then obtain $\mathbf{x}$. If $\mathbf{y}$ is a result of a downscaling transformation of the input image, selfVAE is equivalent to the super-resolution as in [27].

Further, we can generate $\mathbf{y}$ instead of using $\mathbf{y}^*$ and choose to sample or infer $\mathbf{z}$. In this way, we can reconstruct an image in two ways. First, we go from $\mathbf{x}^*$ to $\mathbf{u}$, and then go back to $\mathbf{x}$ using a purely generative path (we refer to it as *reconstruction 1*, see Figure 3iv): $\mathbf{y}^* = d(\mathbf{x}^*) \rightarrow \mathbf{u} \sim q(\mathbf{u}|\mathbf{y}^*) \rightarrow \mathbf{y} \sim p(\mathbf{y}|\mathbf{u}) \rightarrow \mathbf{z} \sim p(\mathbf{z}|\mathbf{u},\mathbf{y}) \rightarrow \mathbf{x} \sim p(\mathbf{x}|\mathbf{z},\mathbf{y})$. Alternatively, we can use the variational posterior to sample $\mathbf{z}$ as in the following procedure (we refer to it as *reconstruction 2*, see Figure 3v): $\Big(\mathbf{y}^* = d(\mathbf{x}^*) \rightarrow \mathbf{u} \sim q(\mathbf{u}|\mathbf{y}^*) \rightarrow \mathbf{y} \sim p(\mathbf{y}|\mathbf{u})\Big)$, then $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}^*) \rightarrow \mathbf{x} \sim p(\mathbf{x}|\mathbf{y},\mathbf{z})$.

In both reconstructions, we want to obtain $\mathbf{u}$. To understand the purpose, we can again think of compression. Instead of sending high-dimensional variables, we can share the low-dimensional $\mathbf{u}$ and, then, infer all other variables. However, in the reconstruction 2 procedure, we must send $\mathbf{z}$ as well. As we will see in the experiments, this trade-off of the number of sent variables allows us to play with the compression ratio and the quality of the reconstruction.

The presented versions of generating and reconstructing images could be useful in the compression task. As we will see in the experiments, each option creates a different ratio of the reconstruction quality against the memory that we need to allocate to send information. However, every inferred variable needs to be sent; thus, more sampling corresponds to lower memory requirements. We want to highlight that, during training, we sample from variational posteriors, $\mathbf{z}^* \sim q(\mathbf{z}|\mathbf{x}^*), \mathbf{u}^* \sim p(\mathbf{u}|\mathbf{y}^*)$, and $\mathbf{y}^* = d(\mathbf{x}^*)$, and then we use these samples for the generative part and in the ELBO eventually. None of the procedures presented in this subsection are used for training. Here, we outline that the selfVAE allows various options for generation and reconstruction, with potential use for compression.

3.4. Hierarchical Self-Supervised VAE

The proposed approach can be further extended and generalized by introducing multiple transformations. For instance, we can achieve this by applying downscaling multiple times. As a result, by incorporating multiple self-supervised representations of the data, the process of modeling a high-dimensional complex density breaks down into K simpler modeling tasks. The resulting model is a K-level VAE architecture, where the overall expressivity of the model grows even further and gradually leads to generations of higher quality. It is important to realize that some transformations cannot be applied multiple times, e.g., edge detection; however, others could be used sequentially, e.g., downscaling.

Formally, we consider K self-supervised data transformations $d_k(\cdot)$ that result in K representations denoted by $\mathbf{y}_{1:K} = [\mathbf{y}_1, \ldots, \mathbf{y}_K]$, where $\mathbf{y}_k = d_k(\mathbf{y}_{k-1})$ and $\mathbf{y}_0 = \mathbf{x}$. Thus, the joint distribution is factorized as follows:

$$p(\mathbf{x}, \mathbf{y}_{1:K}) = p(\mathbf{x}) \prod_{k=1}^{K} \delta(\mathbf{y}_k - d_k(\mathbf{y}_{k-1})). \tag{20}$$

However, analogously to the model with a single $\mathbf{y}$, we can introduce additional latent variables for each level to compensate for lost information, $\mathbf{z}_k$, and another latent variable on top, $\mathbf{u}$. As a result, our model can be expressed in the following form:

$$\begin{aligned} p(\mathbf{x}, \mathbf{y}_{1:K}, \mathbf{z}_{1:K}, \mathbf{u}) =& p(\mathbf{x}|\mathbf{y}_1, \mathbf{z}_1) \prod_{k=1}^{K-1} \Big\{ p(\mathbf{z}_k|\mathbf{y}_k, \mathbf{z}_{k+1}) \times p(\mathbf{y}_k|\mathbf{y}_{k+1}, \mathbf{z}_{k+1}) \Big\} \times \\ & \times p(\mathbf{z}_K|\mathbf{u}, \mathbf{y}_K)\; p(\mathbf{y}_K|\mathbf{u})\; p(\mathbf{u}). \end{aligned} \tag{21}$$

Further, due to the intractability of the marginal likelihood function, we utilize the following family of variational distributions:

$$Q(\mathbf{u}, \mathbf{z}|\mathbf{x}, \mathbf{y}_{1:K}) = q(\mathbf{u}|\mathbf{y}_K) q(\mathbf{z}_1|\mathbf{x}) \prod_{k=1}^{K-1} q(\mathbf{z}_{k+1}|\mathbf{y}_k). \tag{22}$$

After applying the variational inference, we obtain the following objective:

$$\begin{aligned} \ln p(\mathbf{x}, \mathbf{y}_{1:K}) \geq& \mathbb{E}_Q \Big[\ln p_\theta(\mathbf{x}|\mathbf{y}_1, \mathbf{z}_1) + \sum_{k=1}^{K-1} \Big\{ \ln p(\mathbf{z}_k|\mathbf{y}_k, \mathbf{z}_{k+1}) + \\ & + \ln p(\mathbf{y}_k|\mathbf{y}_{k+1}, \mathbf{z}_{k+1}) \Big\} + \ln p(\mathbf{z}_K|\mathbf{u}, \mathbf{y}_K) + \\ & + \ln p(\mathbf{y}_K|\mathbf{u}) + \ln p(\mathbf{u}) - \ln q(\mathbf{u}|\mathbf{y}_K) + \\ & - \ln q(\mathbf{z}_1|\mathbf{x}) - \sum_{k=1}^{K-1} \ln q(\mathbf{z}_{k+1}|\mathbf{y}_k) \Big]. \end{aligned} \tag{23}$$

4. Experiments

4.1. Experimental Setup

- **Datasets** We evaluated the proposed model on the CIFAR-10, Imagenette64, and CelebA datasets:
  - *CIFAR-10* The CIFAR-10 (https://www.cs.toronto.edu/~kriz/cifar.html (accessed on 11 June 2021)) dataset is a well-known image benchmark data containing 60,000 training natural images and 10,000 validation natural images. Each image is of size 32 px × 32 px. From the training data, we put aside 15% randomly selected images as the test set. We augmented the training data by using random horizontal flips and random affine transformations and normalized the data uniformly in the range (0, 1).
  - *Imagenette64* Imagenette64 (https://github.com/fastai/imagenette (accessed on 11 June 2021)) is a subset of 10 easily classified classes from Imagenet https://www.image-net.org/ (accessed on 11 June 2021) (tench, English springer, cassette player, chain saw, church, French horn, garbage truck, gas pump, golf ball, and parachute). We downscaled the dataset to 64 px × 64 px images. Similarly to CIFAR-10, we put aside 15% randomly selected training images as the test set. We used the same data augmentation as in CIFAR-10. Please note that Imagenette64 is **not** equivalent to Imagenet64.
  - *CelebA* The Large-scale CelebFaces Attributes (CelebA) (http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html (accessed on 11 June 2021)) dataset consists of 202, 599 images of celebrities. We cropped the original images on the 40 vertical and 15 horizontal components of the top left corner of the crop box, where the height and width were cropped to 148. In addition to the uniform normalization of the image, no other augmentation was applied.
- **Architectures** The encoders and decoders consisted of building blocks composed of DenseNets [30], channel-wise attention [31], and ELUs [32] as activation functions. The dimensionality of all the latent variables was kept at $8 \times 8 \times 16 = 1024$, and all models were trained using AdaMax [33] with data-dependent initialization [34]. Regarding the selfVAEs, in CIFAR-10, we used an architecture with a single downscaled transformation (selfVAE-downscale), while, on the remaining two datasets (CelebA and Imagenette64), we used a hierarchical three-leveled selfVAE with downscaling, and a selfVAE with sketching. Note that using sketching multiple times would result in no new data representation.
  All models were employed with the bijective prior (RealNVP). All models were comparable in terms of the number of parameters (the range of the weights of all models was from 32 million to 42 million). Due to our limited computational resources, these were the largest models that we were able to train. A fair comparison to the current SOTA models that use over 100 million weights [12,13] is, therefore, slightly skewed. For more details on the model architectures, please refer to the Appendix A.1.
- **Evaluation** We approximated the negative log-likelihood using 512 IW-samples [16] and express the scores in bits per dimension (*bpd*). Additionally, for CIFAR-10, we used the *Fréchet Inception Distance* (FID) [35]. We used the FID score indicating that sometimes the likelihood-based models achieved very low *bpd*; however, their perceptual quality was lower.

4.2. Quantitative Results

We present the results of the experiments on the benchmark datasets in Table 1. First, we notice that, on CIFAR-10, our implementation of the VAE was still lagging behind the other generative models in terms of the *bpd*; however, it was better or comparable in terms of the FID. Surprisingly, the selfVAE-downscale achieved worse *bpd* compared with the VAE with the bijective prior. A possible explanation may lie in the small image size

(32 × 32), as the benefits of breaking down the learning process into two or more steps are not obvious given the small target dimensional space.

Table 1. Quantitative comparison on the test sets from CIFAR-10, CelebA, and Imagenette64. If available, we provide a score measured on the training set (depicted by *). Best results are in bold.

Dataset	Model	*bpd* ↓	FID ↓
CIFAR-10	PixelCNN [36]	3.14	65.93
	GLOW [20]	3.35	65.93
	ResidualFlow [37]	3.28	46.37
	BIVA [12]	3.08	-
	NVAE [13]	2.91	-
	Very deep VAEs [38]	**2.87**	
	DDPM [39]	3.75	**5.24** (**3.17** *)
	VAE (ours)	3.51	41.36 (37.25 *)
	selfVAE-downscale	3.65	34.71 (29.95 *)
CelebA	RealNVP [19]	3.02	-
	VAE (ours)	3.12	-
	selfVAE-sketch	3.24	-
	selfVAE-downscale-3lvl	**2.88**	-
Imagenette64	VAE (ours)	3.85	-
	selfVAE-downscale-3lvl	**3.70**	-

Nevertheless, the selfVAE-downscale achieved significantly better FID scores over the other generative models, except for the diffusion-based generative model [39]. This result could follow from the fact that downscaling allows maintaining context information about the original image, and, as a result, the general coherence is of higher quality. We want to highlight that the diffusion-based generative model proposed in [39] consists of 1000 layers; thus, comparing it to our approach is not entirely fair.

Interestingly, on the two other datasets, CelebA and Imagenette64, a three-level selfVAE-downscale achieved better *bpd* scores compared with the VAE with the bijective prior. This indicates the benefit of employing a multi-leveled self-supervised framework against the VAE in higher-dimensional data, where the plain model fails to scale efficiently. It appears that the hierarchical structure of self-supervised random variables allows encoding the missing information more efficiently in $\mathbf{z}_k$, in contrast to the vanilla VAE, where all information about images must be coded in $\mathbf{z}$. This result is promising and indicates that the proposed approach is of great potential for generative modeling.

Lastly, comparing the selfVAE-sketch against the hierarchical selfVAE-downscale on CelebA, we clearly see that, in terms of the *bpd*, the hierarchical selfVAE-downscale was significantly better. The selfVAE-sketch achieved an even worse *bpd* than the VAE and RealNVP. However, the generations and reconstructions (see Figures 4 and A4) indicate that learning with $\mathbf{y}$ allowed obtaining crisp images. The only issue was learning a proper color tone (e.g., see Figure A4).

4.3. Qualitative Results

We present generations on CelebA in Figure 4 and on CIFAR-10 and Imagenette64 in Figure A2 (see the Appendix), and reconstructions on CIFAR-10 & CelebA in Figure 5.

We first notice that the generations from selfVAE seem to be more coherent in contrast with these from VAE, which produced, overall, more contextless and distorted generations. This result appears to be in line with the FID scores. Especially for CelebA, we observed impressive synthesis quality, great sampling diversity, and coherent generations (Figure 4). On the Imagenette64 dataset, we also observed crisper generations for our method compared to the VAE (see Figure A2 in the Appendix A.3).

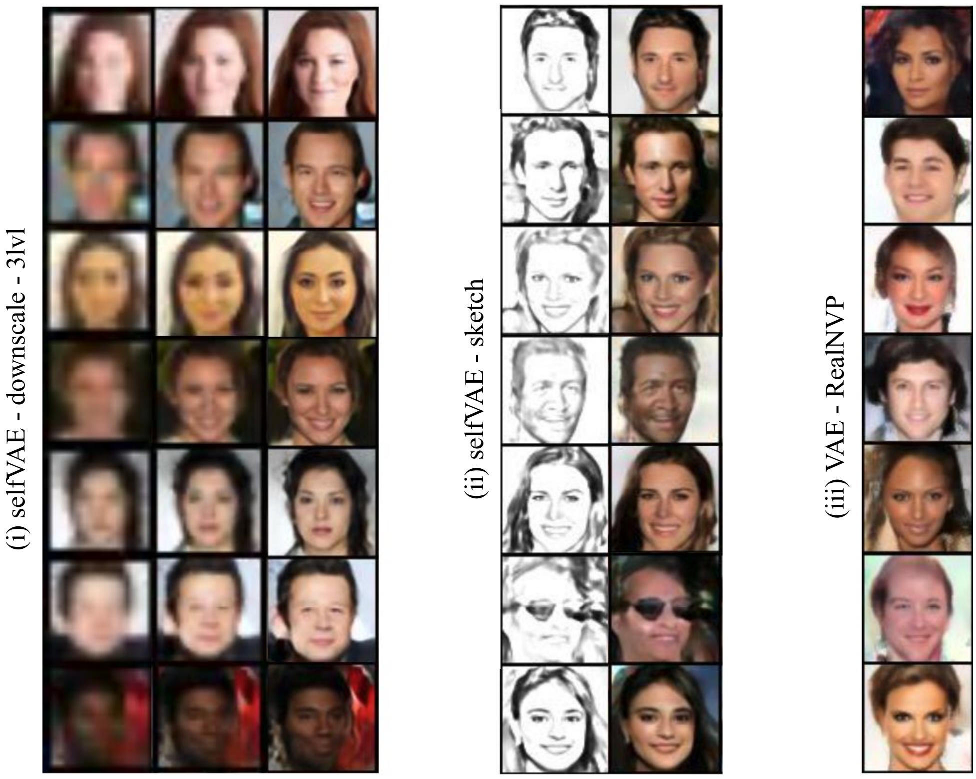


Figure 4. Unconditional CelebA generations from (**i**) the three-level self-supervised VAE with downscaling, (**ii**) the self-supervised VAE employed with edge detection (sketches), and (**iii**) the VAE with the RealNVP prior.

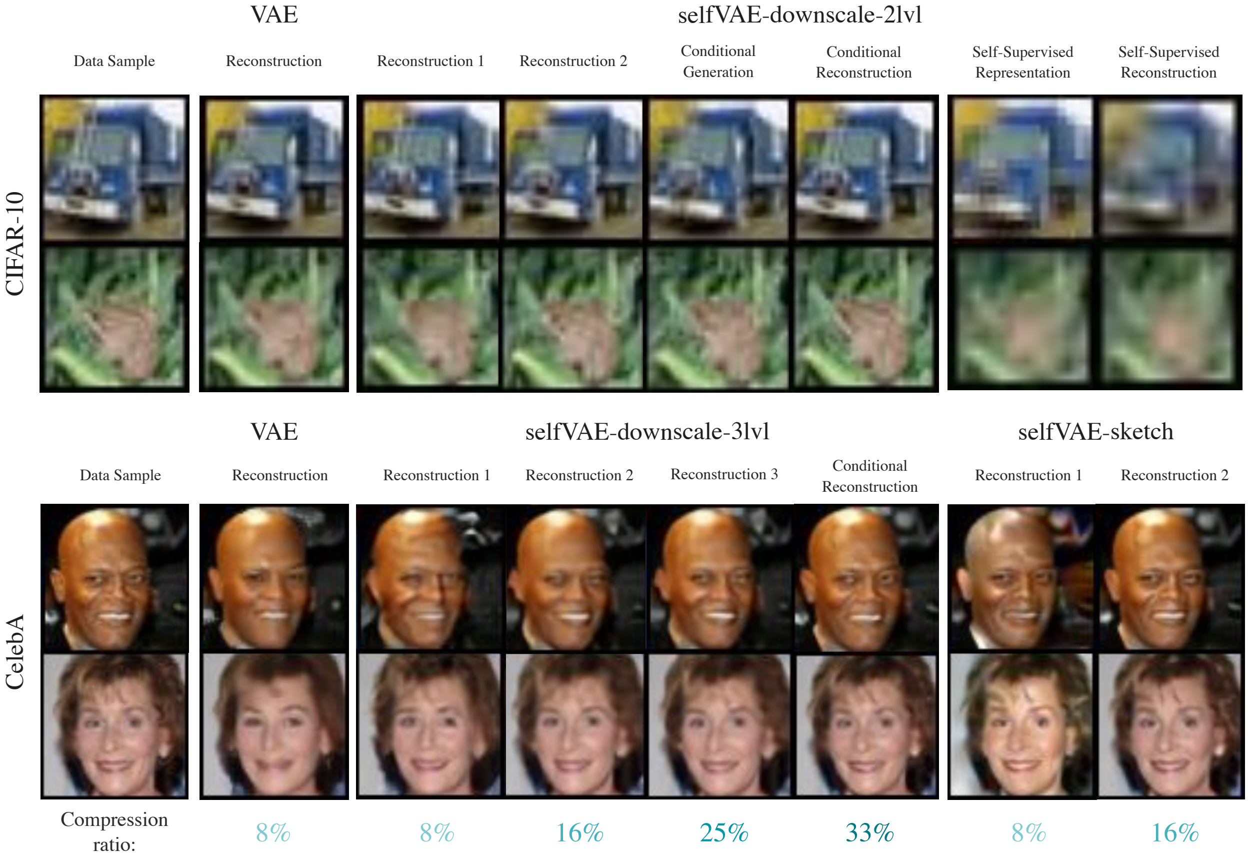
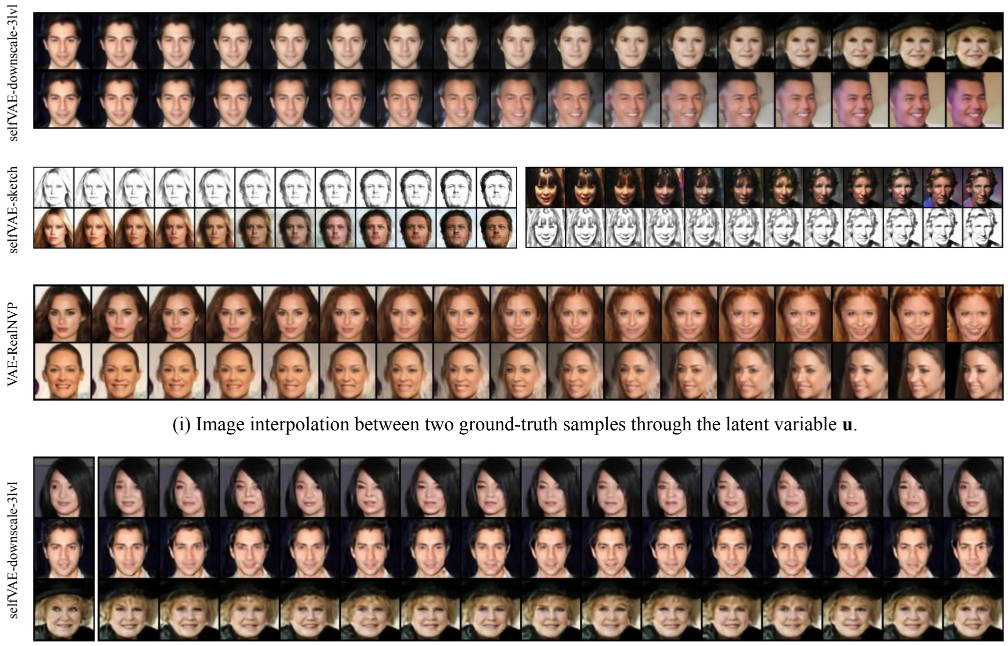


Figure 5. Comparison of image reconstructions with different amounts of sent information.

Furthermore, the hierarchical selfVAE seems to be of great potential for compression purposes. In contrast to the VAE, which is restricted to using a single way of reconstructing an image, the selfVAE allows four various options with different quality/memory ratios (Figure 5). In the selfVAE-sketch, we can retrieve an image with high accuracy by using only 16% of the original data, as it manages to encode all the texture of the image to **z** (Figure A3). This shows the advantage of choosing prior knowledge in the learning process. Lastly, the latent variables added extra information, which defined the result, and we can alter the details of an image, such as the facial expressions (Figure 6ii, see the Appendix A.3).



(i) Image interpolation between two ground-truth samples through the latent variable **u**.

(ii) Image generation given the latent variable **u** and sampling the latent codes $\mathbf{z}_1$ and $\mathbf{z}_2$.

Figure 6. Latent space interpolations and conditional generations of the selfVAEs.

In Figure 6, we visualize (i) interpolations through two ground-truth images through the latent code **u** and (ii) image reconstructions, where we keep the latent code **u** but vary all the others ($\mathbf{z}_1$ and $\mathbf{z}_2$) of the three-leveled selfVAE architecture. As with the previous cases, in the first case, we see that the model incorporated a rich latent space **u**, which is responsible for the generation and construction of the global structure of the image.

Moving from one latent code **u** of a given image to another, we obtained meaningful modifications of the image that resulted in images that share characteristics from both of them. However, in the latter case, we see that we can alter only the high-level features of the image when we keep the values of **u** but vary the others: $\mathbf{z}_1$ and $\mathbf{z}_2$. Interestingly, we see that, given the ground-truth image that is illustrated on the very left, we can sample different expressions and characteristics of the same person, as the latent variable **u** is kept constant.

5. Conclusions

In this paper, we showed that taking deterministic and discrete transformations resulted in coherent generations of high visual quality, and allowed integrating the prior knowledge without losing its unconditional generative functionality. The experimental results confirm that the hierarchical architectures performed better and allowed obtaining better *bpd* scores as well as better generations and reconstructions.

In the experiments, we considered two classes of image transformations, namely, *downscaling* and edge detection (*sketching*). However, there are many possible other transformations (see Figure 1), and we leave investigating them for future work. Moreover, we find the proposed approach interesting for the compression task. A similar approach with a multi-scale auto-encoder for image compression was proposed, e.g., by [40,41]. However, we still used a probabilistic framework and indicate that various non-trainable image transformations (not only multiple scales) could be of great potential.

Author Contributions: Conceptualization, J.M.T.; methodology, I.G. and J.M.T.; software, I.G.; validation, I.G.; investigation, I.G. and J.M.T.; writing—original draft preparation, I.G. and J.M.T.; writing—review and editing, I.G. and J.M.T.; supervision, J.M.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank Maarten Stol (BrainCreators) and Efstratios Gavves (University of Amsterdam) for their support and fruitful discussions.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Appendix A.1. Neural Network Architecture

For the building blocks of the network, we employed densely connected convolutional networks. Additionally, exponential linear units (ELUs) were used everywhere as activation functions. Typically, every convolution operation preceded a batch normalization layer. We instead used weight normalization, where, even though it separates the weight vector from its direction as with batch normalization, it does not make use of the variance. In addition, we used a data-dependent initialization of the model parameters, by sampling the first batch of the training set. This will allow the parameters to be adjusted by taking into account the output of the previous layers and, thus, resulting in a faster learning process.

An important element of the auto-encoding scheme is the process of feature downscaling and upscaling. For the upscaling operation, even though various methods have been proposed [42], we found that the plain transposed convolution generalized better than the others while requiring far less trainable parameters. Finally, inspired by the recent advantages of super-resolution neural network architectures, we used *channel-wise attention* blocks (CA) at the end of every DenseNet block [31]. The CA blocks will help the network to focus on more informative features by exploiting the inter-dependencies among feature channels.

The core building blocks and the network of an auto-encoding network are illustrated in Figure A1.

Appendix A.2. Self-Supervised VAE—Unconditional Generations

Unconditional samples for Imagenette64 and CIFAR-10 are shown in Figure A2.

Appendix A.3. Self-Supervised VAE—Sketch Reconstructions

Given the astonishing performance on visual tasks, CNNs are commonly thought to recognize objects by learning increasingly complex representations of object shapes. However, ref [43] showed that where humans see shapes, CNNs are strongly biased towards recognizing textures. Furthermore, architectures that learn shape-based representations come with several unexpected emergent benefits, such as previously unseen robustness toward a wide range of image distortions. This acted as motivation to employ the framework of a self-supervised auto-encoder with a representation that captured the shape of the object. Thus, the first part models the outlines of the given object, while the second part is responsible for its texture.

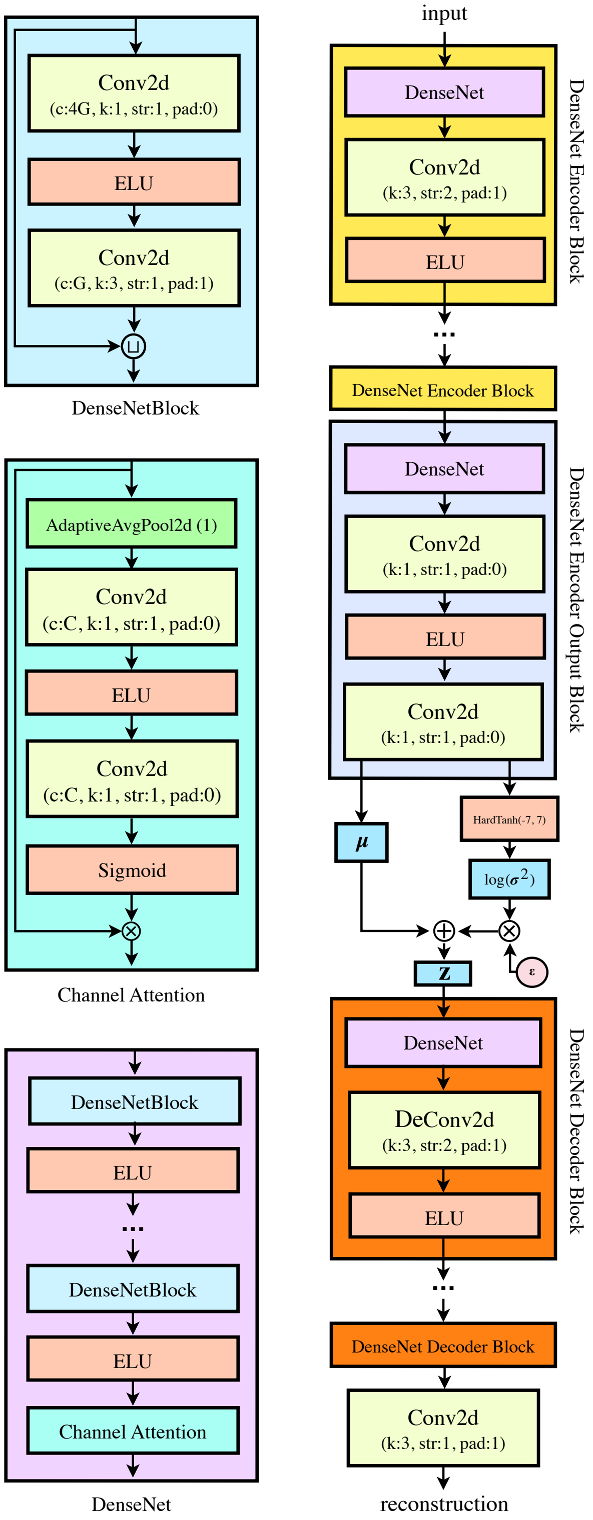


Figure A1. The architecture of our autoencoder. (**left**) The basic building blocks of the network. The notation as 'G' on the Conv2D channels indicates the growth rate of the densely connected network. (**right**) The VAE architecture. The ϵ indicates a random variable drawn from a standard Gaussian, which helps us to make use of the *reparametrization* trick. Until **z**, we refer to this architecture as *Encoder NN* and, thereafter, as *Decoder NN*. *Notation:* c—the number of channels, k—the kernel size, str—the stride size, and pad—the pad size.

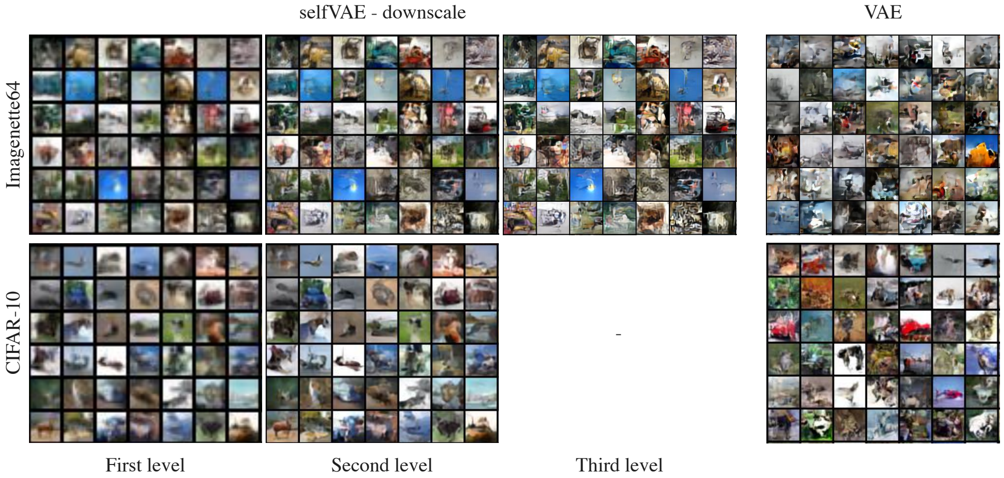


Figure A2. Unconditional generations on Imagenette64 and CIFAR-10.

We can retrieve a shape-based representation of an image by detecting its edges. Edges appear when there is a sharp change in brightness and it usually corresponds to the boundaries of an object. There are many different techniques for computing the edges, like using filters that extract the gradient of the image (Sobel kernels, Laplacian, Gaussian, etc.). In this experiment, we used the method proposed in [44], as it is a fast, high-quality, edge-preserving filtering of images. Specifically, in order to obtain a pencil *sketch* (that is, a black-and-white drawing) of the input image, we made use of two image-blending techniques, known as *dodging* and *burning*. The former lightens an image, whereas the latter darkens it. A sketch transformation can be obtained by using dodge to blend the grayscale image with its blurred inverse. In this way, we produce high-quality edge-preserving filtering of the image efficiently performed in linear time.

Qualitative Analysis

Similar to the case when we used a downscaled transformation of the input image as conditional representation, the selfVAE framework here also allows for different ways to reconstruct an image. The qualitative results when we employ a sketch representation are visible in Figure A3. To start with, we see that, even though, due to the sketch representation, we lose the texture of the image, we preserve not only the global information but also the high-level details that characterize each person. In this way, we let the generative model emphasize these specifics at its first step, which, through the latent variable $\mathbf{u}$, manages to reconstruct with tremendous accuracy.

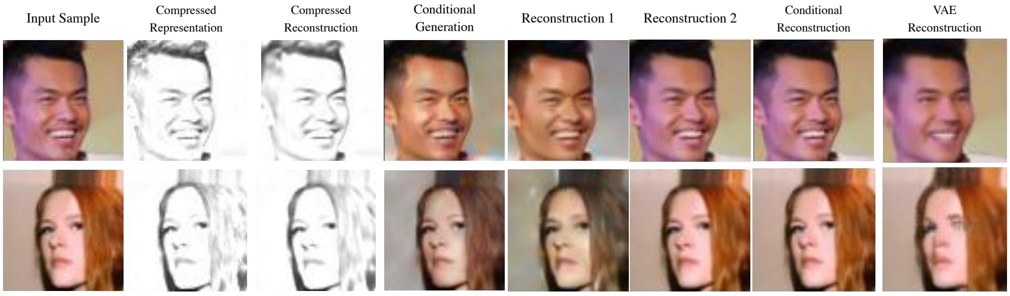


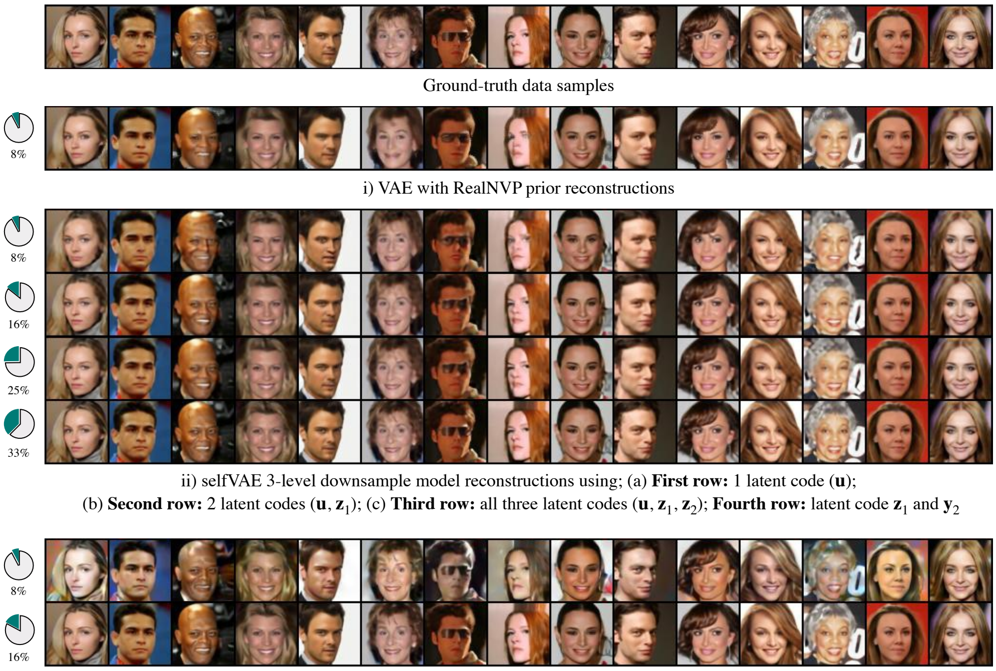
Figure A3. Qualitative results illustrating all the reconstruction techniques on CelebA for selfVAE-sketch.

This can be confirmed by visually comparing the images of the original compressed images (CM) with those of the reconstructed ones. These great results hold also for the conditional reconstruction, where the original sample (OG) is synthesized from conditioning on the original sketch representation (CM). Furthermore, we gain further interpretation of the model through the reconstructions that use only the latent variable $\mathbf{u}$ (RS1) and both latent variables (RS2). Given that both methods infer the sketch image, the results still preserve all detail concepts of the human portrait. However, they are different in terms of the texture (color) of the image, which is modeled through $\mathbf{z}$.

On the one hand, in the first case, its value is inferred through the sketch, and thus it produces the most probable values. That is why we see that, on ground-truth images that incorporate unusual lighting, it outputs a more natural outcome. On the other hand, in the second case, the latent codes of $\mathbf{z}$ are computed given the original sample. From the results, we can see that, indeed, all the information about the texture of the image can be compressed into the latent $\mathbf{z}$, as these reconstructions are identical with the input ones. Additionally, the compressed generation (CG) process is similar to this of RS1, as they both infer the texture, but different, as it uses the ground-truth sketch representation. It is worth noting that the results in both cases share the same quality, which is another indication that the reconstructions of the sketch images are excellent.

Appendix A.4. Additional Results

Additional results of reconstructions for CelebA are shown in Figure A4.



Ground-truth data samples

i) VAE with RealNVP prior reconstructions

ii) selfVAE 3-level downsample model reconstructions using; (a) **First row:** 1 latent code ($\mathbf{u}$); (b) **Second row:** 2 latent codes ($\mathbf{u}, \mathbf{z}_1$); (c) **Third row:** all three latent codes ($\mathbf{u}, \mathbf{z}_1, \mathbf{z}_2$); **Fourth row:** latent code $\mathbf{z}_1$ and $\mathbf{y}_2$

iii) selfVAE 2-level sketch model reconstructions using; (a) **First row:** 1 latent code ($\mathbf{u}$); (b) **Second row:** 2 latent codes ($\mathbf{u}, \mathbf{z}_2$).

Figure A4. Comparison on image reconstructions with different amounts of sent information.

References

1. Rezende, D.J.; Mohamed, S.; Wierstra, D. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *arXiv* **2014**, arXiv:1401.4082.
2. van den Berg, R.; Hasenclever, L.; Tomczak, J.M.; Welling, M. Sylvester Normalizing Flows for Variational Inference. *arXiv* **2018**, arXiv:1803.05649.
3. Hoogeboom, E.; Satorras, V.G.; Tomczak, J.M.; Welling, M. The Convolution Exponential and Generalized Sylvester Flows. *arXiv* **2020**, arXiv:2006.01910.
4. Maaløe, L.; Sønderby, C.K.; Sønderby, S.K.; Winther, O. Auxiliary Deep Generative Models. *arXiv* **2016**, arXiv:1602.05473.
5. Chen, X.; Kingma, D.P.; Salimans, T.; Duan, Y.; Dhariwal, P.; Schulman, J.; Sutskever, I.; Abbeel, P. Variational Lossy Autoencoder. *arXiv* **2016**, arXiv:1611.02731.
6. Habibian, A.; Rozendaal, T.V.; Tomczak, J.M.; Cohen, T.S. Video compression with rate-distortion autoencoders. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 7033–7042.
7. Lavda, F.; Gregorová, M.; Kalousis, A. Data-dependent conditional priors for unsupervised learning of multimodal data. *Entropy* **2020**, *22*, 888. [CrossRef]

8. Lin, S.; Clark, R. LaDDer: Latent Data Distribution Modelling with a Generative Prior. *arXiv* **2020**, arXiv:2009.00088.
9. Tomczak, J.M.; Welling, M. VAE with a VampPrior. *arXiv* **2017**, arXiv:1705.07120.
10. Gulrajani, I.; Kumar, K.; Ahmed, F.; Taiga, A.A.; Visin, F.; Vazquez, D.; Courville, A. PixelVAE: A Latent Variable Model for Natural Images. *arXiv* **2016**, arXiv:1611.05013.
11. Zhao, S.; Song, J.; Ermon, S. Learning hierarchical features from deep generative models. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 4091–4099.
12. Maaløe, L.; Fraccaro, M.; Liévin, V.; Winther, O. BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. *arXiv* **2019**, arXiv:1902.02102.
13. Vahdat, A.; Kautz, J. NVAE: A Deep Hierarchical Variational Autoencoder. *arXiv* **2020**, arXiv:2007.03898.
14. Jordan, M.I.; Ghahramani, Z.; Jaakkola, T.S.; Saul, L.K. An introduction to variational methods for graphical models. In *Machine Learning*; MIT Press: Cambridge, MA, USA, 1999; pp. 183–233.
15. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. *arXiv* **2013**, arXiv:1312.6114.
16. Burda, Y.; Grosse, R.; Salakhutdinov, R. Importance Weighted Autoencoders. *arXiv* **2015**, arXiv:1509.00519.
17. Hoffman, M.D.; Johnson, M.J. Elbo surgery: Yet another way to carve up the variational evidence lower bound. In Proceedings of the NIPS 2016 Workshop on Advances in Approximate Bayesian Inference, Barcelona, Spain, 9 December 2016; Volume 1, p. 2.
18. Rosca, M.; Lakshminarayanan, B.; Mohamed, S. Distribution Matching in Variational Inference. *arXiv* **2018**, arXiv:1802.06847.
19. Dinh, L.; Sohl-Dickstein, J.; Bengio, S. Density estimation using Real NVP. *arXiv* **2016**, arXiv:1605.08803.
20. Kingma, D.P.; Dhariwal, P. Glow: Generative Flow with Invertible 1x1 Convolutions. *arXiv* **2018**, arXiv:1807.03039.
21. Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P.A. Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 1096–1103.
22. Zhang, R.; Isola, P.; Efros, A.A. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1058–1067.
23. Hénaff, O.J.; Srinivas, A.; De Fauw, J.; Razavi, A.; Doersch, C.; Eslami, S.; Oord, A.V.d. Data-efficient image recognition with contrastive predictive coding. *arXiv* **2019**, arXiv:1905.09272.
24. Oord, A.V.d.; Li, Y.; Vinyals, O. Representation learning with contrastive predictive coding. *arXiv* **2018**, arXiv:1807.03748.
25. Liu, X.; Zhang, F.; Hou, Z.; Wang, Z.; Mian, L.; Zhang, J.; Tang, J. Self-supervised Learning: Generative or Contrastive. *arXiv* **2020**, arXiv:2006.08218.
26. Chang, H.; Yeung, D.Y.; Xiong, Y. Super-resolution through neighbor embedding. In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2004, Washington, DC, USA, 27 June–2 July 2004; Volume 1, p. I.
27. Gatopoulos, I.; Stol, M.; Tomczak, J.M. Super-resolution Variational Auto-Encoders. *arXiv* **2020**, arXiv:2006.05218.
28. Chakraborty, S.; Chakravarty, D. A new discrete probability distribution with integer support on $(-\infty,\infty)$. *Commun. Stat. Theory Methods* **2016**, *45*, 492–505. [CrossRef]
29. Salimans, T.; Karpathy, A.; Chen, X.; Kingma, D.P. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications. *arXiv* **2017**, arXiv:1701.05517.
30. Huang, G.; Liu, Z.; van der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. *arXiv* **2016**, arXiv:1608.06993.
31. Zhang, Y.; Li, K.; Li, K.; Wang, L.; Zhong, B.; Fu, Y. Image Super-Resolution Using Very Deep Residual Channel Attention Networks. *arXiv* **2018**, arXiv:1807.02758.
32. Clevert, D.A.; Unterthiner, T.; Hochreiter, S. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *arXiv* **2015**, arXiv:1511.07289.
33. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
34. Salimans, T.; Kingma, D.P. Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. *arXiv* **2016**, arXiv:1602.07868.
35. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *arXiv* **2017**, arXiv:1706.08500.
36. van den Oord, A.; Kalchbrenner, N.; Kavukcuoglu, K. Pixel Recurrent Neural Networks. *arXiv* **2016**, arXiv:1601.06759.
37. Chen, R.T.Q.; Behrmann, J.; Duvenaud, D.; Jacobsen, J.H. Residual Flows for Invertible Generative Modeling. *arXiv* **2019**, arXiv:1906.02735.
38. Child, R. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. *arXiv* **2020**, arXiv:2011.10650.
39. Ho, J.; Jain, A.; Abbeel, P. Denoising diffusion probabilistic models. *arXiv* **2020**, arXiv:2006.11239.
40. Mentzer, F.; Agustsson, E.; Tschannen, M.; Timofte, R.; Gool, L.V. Practical full resolution learned lossless image compression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–21 January 2019; pp. 10629–10638.
41. Razavi, A.; van den Oord, A.; Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2019; pp. 14866–14876.
42. Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A.P.; Bishop, R.; Rueckert, D.; Wang, Z. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. *arXiv* **2016**, arXiv:1609.05158.

43. Geirhos, R.; Rubisch, P.; Michaelis, C.; Bethge, M.; Wichmann, F.A.; Brendel, W. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
44. Gastal, E.S.L.; Oliveira, M.M. Domain Transform for Edge-Aware Image and Video Processing. *ACM TOG* **2011**, *30*, 1–12. [CrossRef]