

## A APPENDIX

### A.1 THE IMPACT OF THE BIJECTIVE PRIOR ON VAES

#### A.1.1 MOTIVATION

Bijective neural networks were first introduced to increase the expressibility of the encoder, due to its forced limitations to follow a mean-field variational family of Gaussian distributions. As they demand the base function to be known and simple (i.e. a distribution that is easy to estimate and sample from), the mapping of the input space to the latent follows a two step process; first the input sample is mapped to the base distribution through the encoder, and then transformed to match the unimodal Gaussian prior. However, despite the richer variational posterior, Rosca et al. (2018) proved that the optimal prior may still not much a fixed unit Gaussian distribution, which in addition, is known to result in over-regularized models that tend to ignore more of the latent codes (Burda et al., 2015; Tomczak & Welling, 2017).

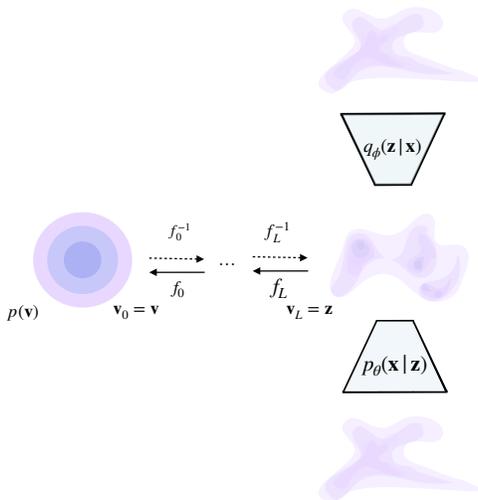


Figure 7: VAE employed with bijective prior.

However, little attention has been paid to adapt bijective neural networks as a learnable, data-driven prior of Variational Auto-Encoders. Since the prior of the latent dimensions plays a crucial part in the objective function, it will contribute into a better likelihood estimation of the data, while providing more informative latent codes. There are a couple of advantages using a bijective network to estimate the prior distribution:

- **Arbitrary complex, data-driven prior** Instead of forcing the encoder to match a fixed distribution, we follow the intuitive process, that now, the prior distribution adapts to the posterior during the training progress. Thus, the posterior is not being dragged to the centre of a standard Gaussian distribution and it is allowed to move freely in the latent space. As the normalizing flows framework allows the transformation of a simple function to an arbitrary complex one, the result would be a richer, multi-modal distribution that incorporates causal knowledge of the data at hand.
- **Simple implementation and integration** Instead of breaking the encoder into a two step process, we fit a flow-based generative model to the latent codes produced by the encoder. Additionally, getting advantage of the invertible nature of the network, we can generate a datapoint by simply sampling a point from its base distribution, and transforming it (through the inverse process) into a latent code of the variational posterior.

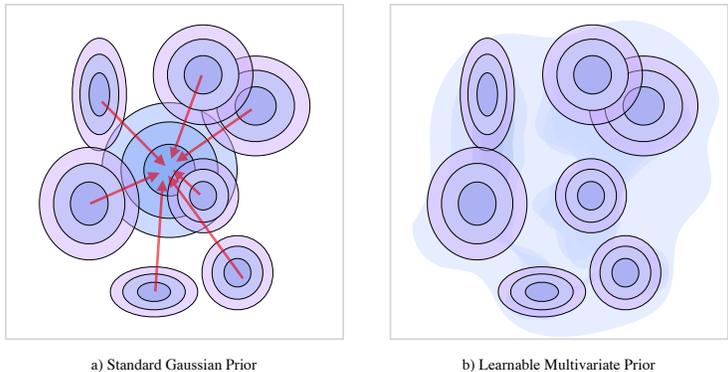


Figure 8: Left: The standard prior is too strong and over-regularizes the encoder and create "holes" in the latent space. Right: Learnable prior adjusts to the posterior distribution, surrounding the variable posterior with smooth transitions.

- **Fast sampling and inference.** An alternative way of retrieving a powerful latent prior is through autoregressive modelling of the latent codes. Formally,  $p(\mathbf{z})$  is factorized as

$$p(\mathbf{z}) = \prod_{i=1}^d p(z_i | \mathbf{z}_{<i})$$

so that estimating  $p(\mathbf{z})$  reduces to the estimation of each single conditional probability density (CPD) expressed as  $p(z_i | \mathbf{z}_{<i})$ , where the symbol  $<$  implies an order over random variables. However, because high-dimensional data often requires a relatively high-dimensional latent space, in order to extract the highly informative causal components, the sampling process can be an important barrier for fast generation of novel content. On the contrary, the bijective neural networks require only a forward pass of a sampled point, reducing significantly the inference time.

It is straightforward to obtain a rich, multi-modal prior, and the process its described on section 2.2.

### A.1.2 EXPERIMENTS

On this section, we will evaluate the performance of Variational Auto-Encoder with using different latent variable prior distributions, both on a quantitatively and qualitatively view. The models named as "VAE" represents the plain architecture with standard normal prior, while "VAE + MoG" and "VAE + RealNVP" represent data-driven priors, where the former employs 10 mixture of Gaussians and the latter the bijective network RealNVP. For fair comparison, all the models share the same neural networks architectures both on the encoder and decoder. For the specific architecture of the neural networks, please refer to the section .

**Quantitative Analysis** Table 2 represents the quantitative results of density estimation on the natural image dataset CIFAR-10. We measure performance by estimating the log-likelihood with 512 importance weighted samples on the test set. To begin with, the use of a mixture of Gaussians, a learned distribution, performs better than the VAE with fix prior, with a negligible cost on additional trainable parameters, which do not allocate extra time in sample generation. However, the adaption of RealNVP as a bijective network, improves significantly the likelihood score. Interestingly, both models with learned priors perform better than the fixed one, by reaching better reconstruction loss with the cost of allocating more regularization cost. This behavior exactly reflects the relationship between variational posterior and the prior; when a fix unimodal Gaussian is used, it pulls the variational ones to its pick. As a result, all the variational posteriors are close to the mean of the prior, resulting into a smaller KL diverge but blurry reconstructions. On the contrary, the learned multi-modal priors allow the posteriors to move more freely and are able to adjust to the variational family (see figure 8). This distribution mismatch is the greatest when the RealNVP prior is used, indicating that the prior allows the encoder to form more and more complex distributions, by trading of regularization loss for better reconstructions.

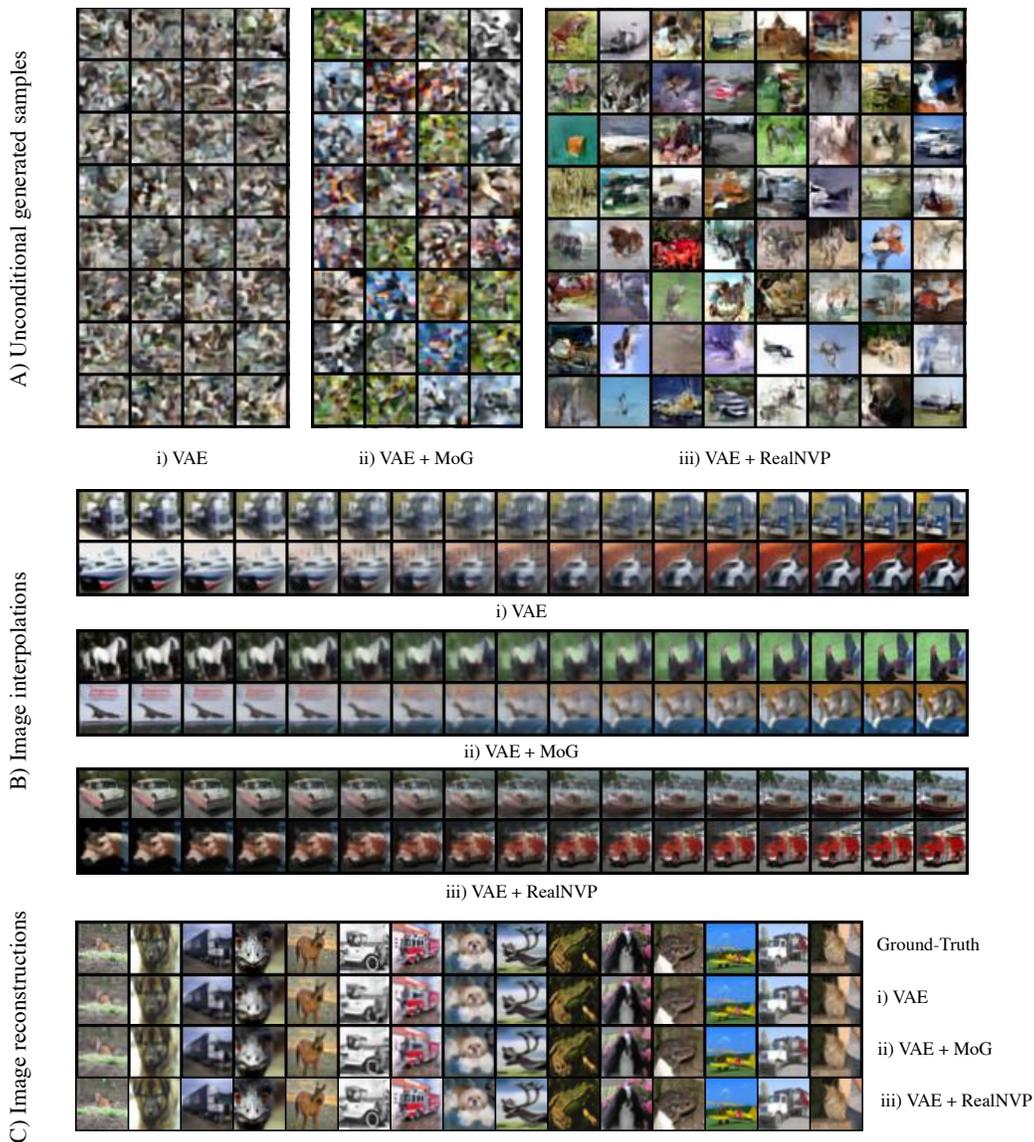


Figure 9: Qualitatively results on CIFAR-10 for VAEs with various latent prior distributions. The depicted results show A) unconditional generative samples B) image interpolations and C) image reconstructions.

**Qualitative Analysis** The difference in performance becomes more obvious by looking at the qualitatively results, presented in Figure 9. Looking at the generative samples, we see a clear improvement when moving to richer prior. The plain VAE with standard normal distribution generates random patterns while the model with a mixture of Gaussian prior enhances them with colors. However, only VAE with RealNVP showcases generations that manage to display a global context. To examine the richness of the learnt space, we also perform interpolation on the latent codes between two ground-truth images. Ideally, the internal generations should result into meaningful modifications of the provided samples. In the case of VAE, we see that the inner generations produce blurry samples, indicating the unexpressivity of the latent codes. The generations are getting better with the MoG prior, but the most impressive results are coming from the framework with RealNVP. We can see that the samples are more influenced by their closest ground-truth image, while adapting more and more core characteristics when moving closer to the other. For example, on the second row of B:iii, the original brown horse first turns into red, adapting the color of the firetruck, before it smoothly results into it. Lastly, on image reconstructions we again witness the same pattern, when the richer

Table 2: Generative modelling performance on CIFAR-10 obtained from different priors in bits per dimension. The generation time is based on 100 runs on a single GeForce GTX 1080 Ti GPU for 1 image (and 100 images).

Model	<i>nll</i>	<i>reconstruction loss</i>	<i>regularization loss</i>	#params	<i>generation time (sec)</i>
	(bits/dim)	RE <sub>x</sub>	KL <sub>z</sub>		
VAE	3.88	6675	1596	20M	0.01 (0.07)
VAE + MoG	3.83	6512	1653	22M	0.01 (0.07)
VAE + RealNVP	<b>3.51</b>	5540	1966	32M	0.07 (0.14)

prior does result into better reconstructions, confirming the quantitatively results presented on table 2. Specifically, the VAE with the bijective prior showcases an excellent performance on the natural image reconstruction task, which is contrary to the performance of the other two approaches. This is associated with the effectiveness of the powerful, invertible, data-driven prior (in our case, the RealNVP) and its ability to boost the performance significantly with negligible sacrifice on generation speed, and none on inference.

## A.2 NEURAL NETWORK ARCHITECTURE

The choice of the NN architecture is crucial for the performance and the scalability of the overall framework, and usually architectures that showcased great performance in discriminate tasks (i.e. classification) are used in generative modelling tasks as well. However, the internal representations that the networks have to discover are fundamentally different, and little attention has been given into designing a NN specifically for an auto-encoder setting. For example, in classification tasks the network extracts specific representation of a particular object, in contrast with the generative models, where we aim for discovering the semantic structure of the data. Thus, as we argue that we can benefit from a carefully designed architecture, in this section we present our approach.

For building blocks of the network, we employed densely connected convolutional networks instead of residual ones. The motivation for this choice is that since DenseNets encourage feature reuse, it will help preserve visual information from the very first layer effectively, while requiring less trainable parameters. Thus, the network could discover easier generic graphical features and local pixel correlations. The concatenation of the filters will also alleviate the vanishing-gradient problem and allow us to build deep architectures. Additionally, exponential linear units (ELUs) are used everywhere as activation functions. In contrast to ReLUs, ELUs have negative values which allows them to push mean unit activations closer to zero, which speeds up the learning process. This is due to a reduced *bias shift effect*; bias that is introduced to the units from those of the previous layer which have a non-zero mean activation.

Typically, every convolution operation precedes a batch normalization layer, as they empirically exhibit a boost in performance in discriminate tasks. However, their performance is known to degrade for small batch sizes, as the variance of the activation noise that they contribute is inversely proportional to the number of data that is processed. This noise injection, in combination with their intensive memory demands, can be critical drawbacks when we process image data, especially high-dimensional ones. We instead use weight normalization, where even though it separates the weight vector from its direction just like batch normalization, they do not make use of the variance. This allows them to get the desired output even in small mini-batches, while allocating a small proportion of memory. We empirically find out that indeed using weight normalization reduces the overfitting of the model. In addition, we used a data-dependant initialization of the model parameters, by sampling the first batch of the training set. This will allow the parameters to be adjusted by the output of the previous layers, taking into account and thus resulting into a faster learning process.

An important element of the auto-encoding scheme is the process of feature downscaling and upscaling. Despite its success in classification tasks, pooling is a fixed operation that replacing it with a stride-convolution layer can also be seen as generalization, as the scaling process is now learned. This will increase the models' expressibility with the cost of adding a negligible amount of learning parameters. For the upscaling operation, even though various methods have been proposed (Shi et al., 2016), we found out that the plain transposed convolution generalised better than the others, while requiring far less trainable parameters. Finally, inspired from the recent advantages on

super-resolution neural network architectures, we used *channel-wise attention* blocks (CA) at the end of every DenseNet block (Zhang et al., 2018). The CA blocks will help the network to focus on more informative features, by exploiting the inter-dependencies among feature channels. Thus, it performs feature recalibration in a global way, where the per-channel summary statistics are calculated and then used to selectively emphasise informative feature-maps as well as suppress useless ones (e.g. redundant feature-maps). This is done through a global average pooling, that squeezes global spatial information into a channel statistical descriptor, followed by a gating mechanism, where it learns nonlinear interactions between the input channels.

The core building blocks and the network of an auto-encoding network are illustrated in Figure 10.

### A.3 SELF-SUPERVISED VAE - SKETCH RECONSTRUCTIONS

Given the astonishing performance on visual tasks, CNNs are commonly thought to recognise objects by learning increasingly complex representations of object shapes. However, Geirhos et al. (2019) showed that where humans see shapes, CNNs are strongly biased towards recognising textures. Furthermore, architectures that learn shape-based representations come with several unexpected emergent benefits such as previously unseen robustness towards a wide range of image distortions. This acted as a motivation to employ the framework of self-supervised auto-encoder with a representation that captures the shape of the object. Thus, the first part will model the outlines of the given object while the second will be responsible for its texture.

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 of Gaussian, etc.). In this experiment, we will use the method proposed in Gastal & Oliveira (2011), 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 will make 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 11. To start with, we see that even though because of the sketch representation we lose the texture of the image, we preserve not only the global information but also high-level details that characterize each person. In this way, we let the generative model emphasise on these specifics at its first step, which through the latent variable  $\mathbf{u}$ , manages to reconstruct with tremendous accuracy. 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 synthesised 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 end results still preserve all detail concepts of the human portrait. However, they are different in terms of the texture (colour) of the image, which is modelled through  $\mathbf{z}$ . One 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 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. And 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 alike to this of RS1 as they both infer the texture but different as it uses the ground-truth sketch representation. It is worth to note that the end result in both cases share the same quality, which is another indication that the reconstructions of the sketch images are excellent.

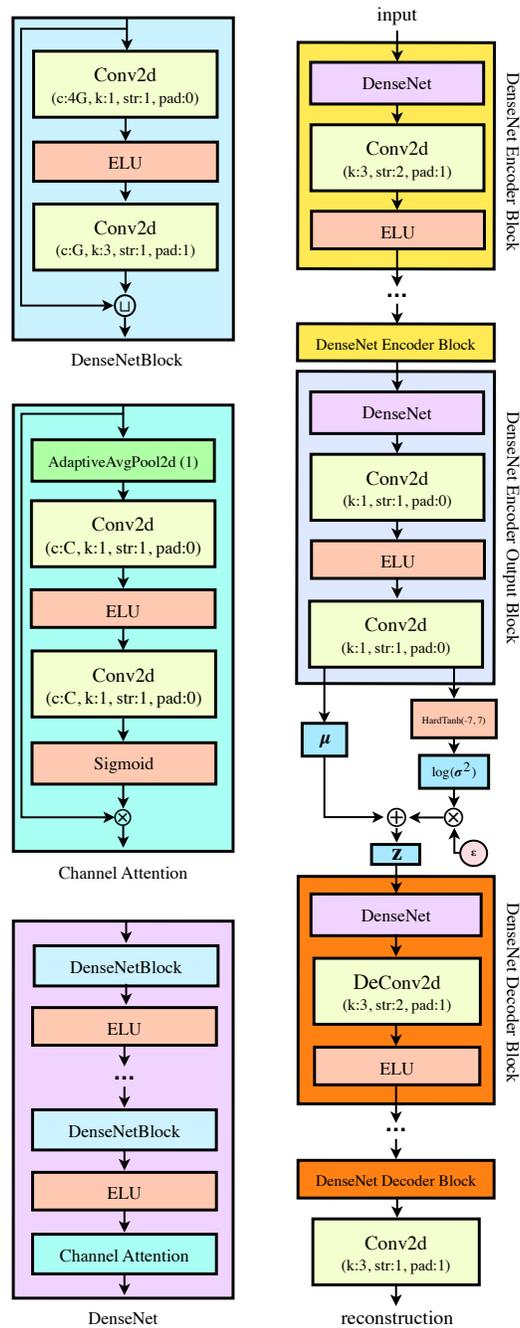


Figure 10: Architecture of our autoencoder. On the right, there are some basic buildings block of the network. The notation as 'G' on the Conv2D channels indicate the growth rate of the densely connected network. The  $\epsilon$  indicates a random variable drawn from a standard Gaussian, which helps us to make use of the *reparametrization* trick. Until  $\mathbf{z}$ , we refer to this architecture as *Encoder NN* and thereafter as *Decoder NN*. The former and the later form the *building blocks* to every model that we train and evaluate.

#### A.4 INTERPOLATION THROUGH LATENT SPACES AND CONDITIONAL GENERATIONS

In Figure 12 we visualise i) interpolations through two ground-truth images through the latent code  $\mathbf{u}$  and ii) image reconstructions, where we keep the latent code  $\mathbf{u}$  but varying all the others ( $\mathbf{z}_1$  and  $\mathbf{z}_2$ )



Figure 11: Qualitative results illustrating all the reconstruction techniques on CelebA for selfVAE-sketch.

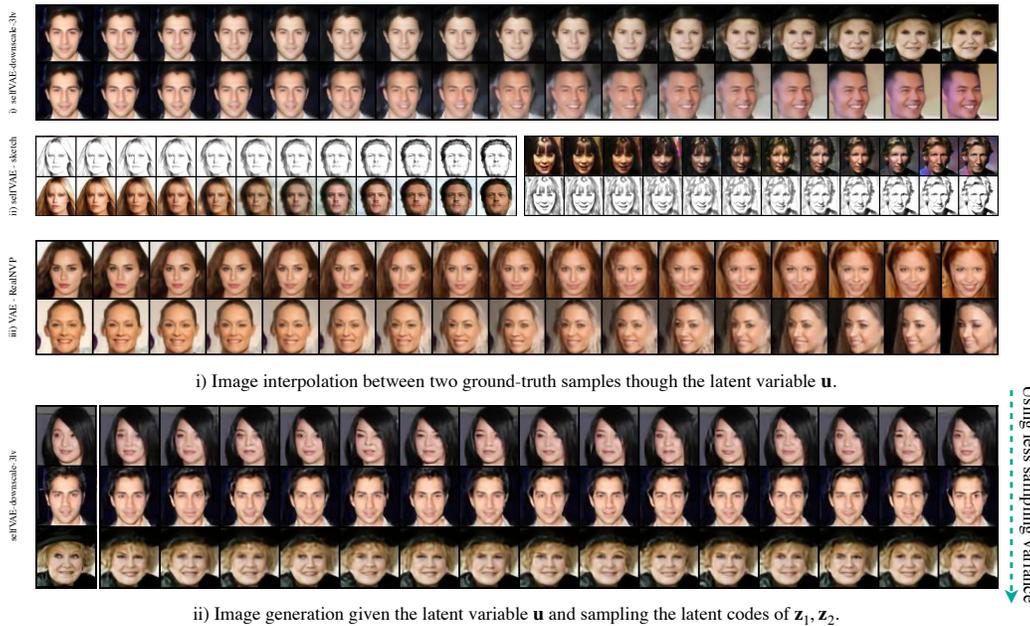


Figure 12: Latent space interpolations and conditional generations of the selfVAEs.

of the 3-leveled selfVAE architecture. Just like the previous cases, in the first case, we see that the model incorporated a rich latent space  $\mathbf{U}$ , which is responsible for the generation and construction of the global structure of the image. Moving from one latent code  $\mathbf{u}$  of a given image to another, we obtain meaningful modifications of the image that result in images that share characteristics from both of them. However, in the latter case, we see that we can alter only high-level features of the image when we keep the values of  $\mathbf{u}$  but vary the others;  $\mathbf{z}_1$  and  $\mathbf{z}_2$ . Interestingly, we see that given that 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  $\mathbf{u}$  is kept constant.

## A.5 ADDITIONAL RESULTS

Additional results of reconstructions for CelebA are shown in Figure 13.

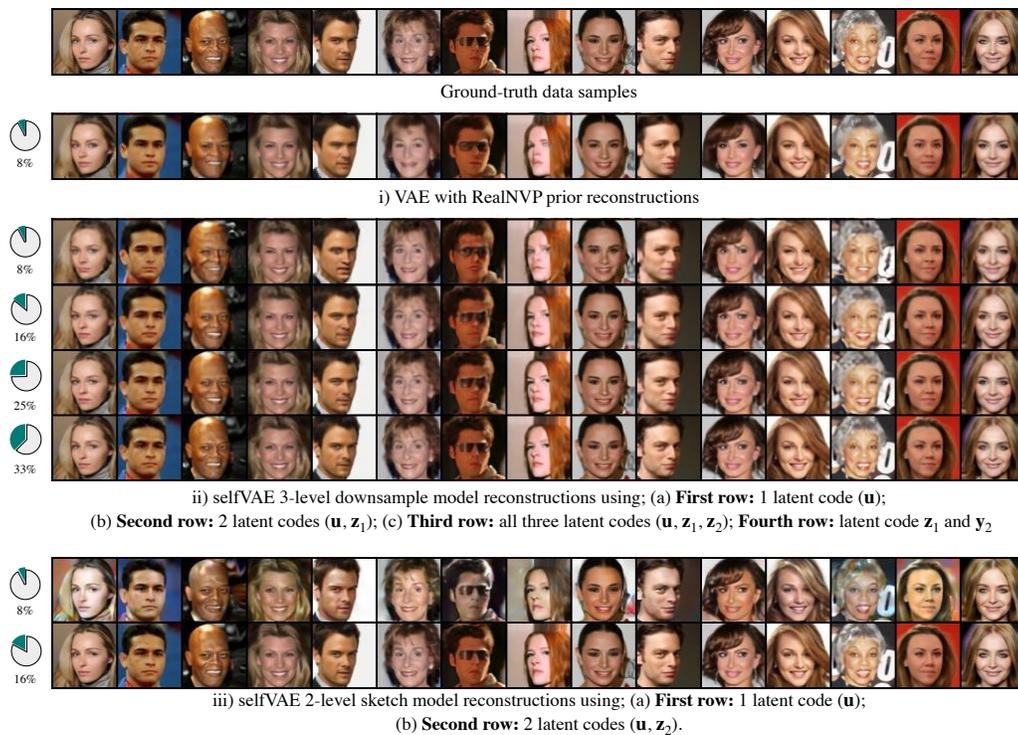


Figure 13: Comparison on image reconstructions with different amount of sent information.