Laplacian Pyramid of Conditional Variational Autoencoders

Garoe Dorta
University of Bath
Anthropics Technology Ltd.
g.dorta.perez@bath.ac.uk

Sara Vicente
Anthropics Technology Ltd.
sara@anthropics.com

Lourdes Agapito
University College London
l.agapito@cs.ucl.ac.uk

Neill D.F. Campbell
University of Bath
n.campbell@bath.ac.uk

Simon Prince
Anthropics Technology Ltd.
simon.prince@anthropics.com

Ivor Simpson
Anthropics Technology Ltd.
ivor@anthropics.com

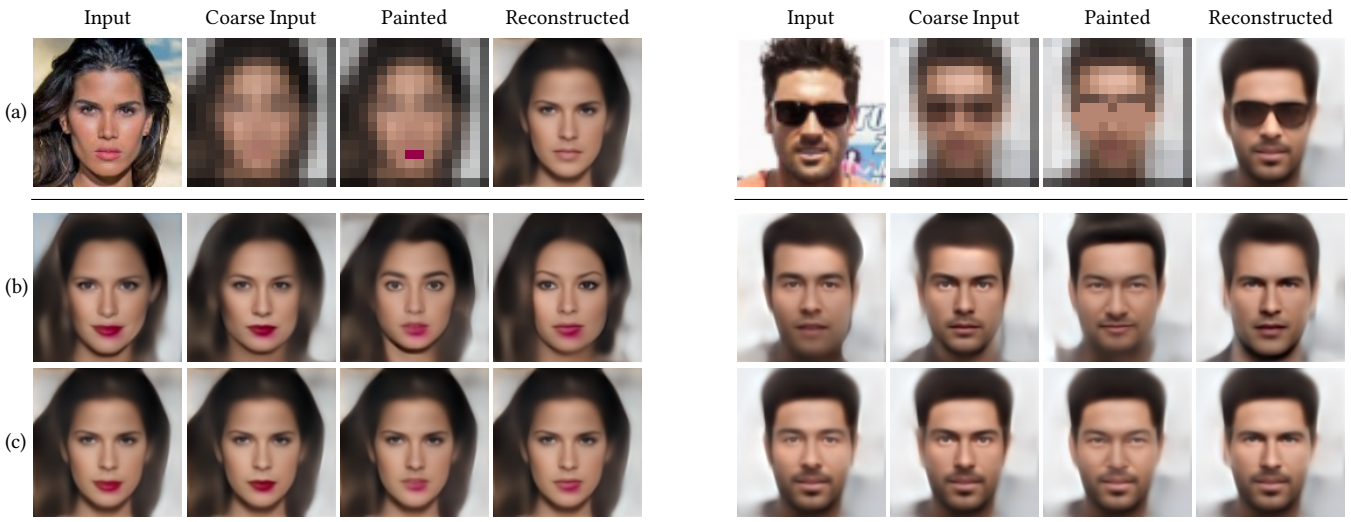

Figure 1: Image editing example. (a) An input image is decomposed using a Laplacian pyramid. The user paints into the coarse image. The input image is projected to a low-dimensional representation and reconstructed from it. (b) Several samples are generated conditioned on the painted coarse image. (c) The regions of interest in (b) are composed with the reconstructed image.

ABSTRACT

Variational Autoencoders (VAE) learn a latent representation of image data that allows natural image generation and manipulation. However, they struggle to generate sharp images. To address this problem, we propose a hierarchy of VAEs analogous to a Laplacian pyramid. Each network models a single pyramid level, and is conditioned on the coarser levels. The Laplacian architecture allows for novel image editing applications that take advantage of the coarse to fine structure of the model. Our method achieves lower reconstruction error in terms of MSE, which is the loss function of the VAE and is not directly minimised in our model. Furthermore, the reconstructions generated by the proposed model are preferred over those from the VAE by human evaluators.

CVMP 2017, December 11–13, 2017, London, United Kingdom

ACM ISBN 978-1-4503-5329-8/17/12…$15.00
https://doi.org/10.1145/3150165.3150172

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; **Neural networks**;

KEYWORDS

Deep Neural Networks, Generative Models, Faces

ACM Reference format:
Garoe Dorta, Sara Vicente, Lourdes Agapito, Neill D.F. Campbell, Simon Prince, and Ivor Simpson. 2017. Laplacian Pyramid of Conditional Variational Autoencoders. In *Proceedings of 14th European Conference on Visual Media Production (CVMP 2017), London, United Kingdom, December 11–13, 2017 (CVMP 2017),* 9 pages.
https://doi.org/10.1145/3150165.3150172

1 INTRODUCTION

General purpose image editing packages such as Adobe Photoshop and Gimp rely on simple image models which have no knowledge of the objects in the scene. Accordingly, extensive knowledge is needed by the user to successfully execute any non-trivial modification. To create more intuitive editing tools, we need more powerful image

models. In this paper we develop a model that contains detailed knowledge of faces.

Intuitive editing controls allow manipulation of *latent attributes*. To model such attributes, we assume that the face image data lies on a low-dimensional manifold embedded in a high-dimensional space. By generating new images such that they lie on the manifold, any point will result in a realistic image. Thus, editing becomes a navigation task in the manifold.

Deep Convolutional Neural Networks (DNN) have recently shown state-of-the-art results in image editing [20, 23] and manifold learning [10]. Autoencoders [3] are trained to learn a non-linear projection of the image data into a low-dimensional latent space. Images can edited by manipulating their representation in the latent space then re-projecting back to image space.

Unfortunately Autoencoder models for image editing have two major drawbacks. First, the generated images tend to be over-smoothed. This effect is especially noticeable in DNNs that use mean squared error metrics which do not correspond to human perception [12, 13]. For example, if an image with high frequency detail is translated by one pixel it can have a large error, but a human might not be able to perceive the difference. Second, scaling these methods to high-resolution images has proven challenging, not least because the number of parameters becomes very large.

We address these problems with the Laplacian Pyramid of Conditional Variational Autoencoders (LapCVAE). This decomposes image generation into a hierarchy of smaller tractable steps. We present a novel loss function that allows the latent Gaussian distributions to have arbitrary mean and variance, but can still be trained and sampled efficiently. We evaluate our model on the CelebA [14] data-set. We show that our network is able to generate compelling images, and that each level in the hierarchy learns a useful representation, as shown in Figure 1.

2 RELATED WORK

There are two common families of deep generative architectures: Variational Autoencoders (VAE) [10] and Generative Adversarial Networks (GAN) [7].

Autoencoders [3] learn an *encoder* that transforms the image data into a latent low-dimensional representation, and a *decoder* that learns the inverse transformation. The model is trained to minimize reconstruction error between the input and decoder output. In the Variational Autoencoder (VAE) [10], the latent space is encouraged to have a hyperspherical structure. Learning the parameters is made tractable by employing a variational approximation for the marginal likelihood of the data. The main drawback of VAEs is that they tend to produce blurry outputs [13, 21]. Some hypothesis for this behavior include, the use of an L_2 loss for the reconstruction error [12], and the assignment of high probability to points not present in the training dataset, in an attempt to generalize when multiple modes are present [6].

Generative Adversarial Networks (GAN) pose image generation as a minmax game, where a generator network is trained to transform vectors of random noise into samples from the input data distribution, and a discriminator network is trained to distinguish between the real data and the samples. Thus, the vector of noise becomes a latent low-dimensional representation of the training data. GAN models are hard to train [17, 19], as the discriminator tends to converge faster than the generator, and they are prone to oscillating behavior. There have been attempts to stabilize the training procedure [17, 19]. Since GAN are only concerned with the generation of plausible outputs, they cope well with data with multiple modes. They can disregard less common modes in order to capture the remaining ones [19]. They typically generate sharper results than the VAE, though training the latter is more straightforward [6]. Hybrid schemes that combine GANs and VAEs have also been proposed [2, 13].

Both of these architectures have been extended using coarse to fine approaches: Zhang et al. [22] divide the GAN architecture into two resolutions. Kolesnikov et al. [11] develop a method based on explicit low to high resolution generation, and Denton et al. [4] build a Laplacian pyramid of GANs. The PixelVAE model [8], extends VAEs by adding PixelCNN [15] layers after the decoder to be able to model very local texture details.

These networks produce latent spaces where the dimensions do not map to human-interpretable variations in the output images. In image editing, it is more intuitive to manipulate semantic attributes. Indeed, a supervised approach to conditioning the output has proved to increase the quality of the results [13, 21]. Conditional GAN [5] add a feature vector, which encodes class labels. A similar approach has been taken for VAEs [21]. Reed at al. [18] included position and orientation as well as class attributes, which are encoded together in a feature vector to control the location and characteristics of the object. However these methods require the conditioning vector to be provided as an input, which makes their general use less practical.

In general, GAN-based approaches are able to produce results of higher quality that VAE-based techniques [6, 13, 21]. However, as shown in Figures 1 and 9 begin able to reconstruct and sample new images is an integral part of our model and its image editing applications. GAN models are designed for sampling only, reconstructing images requires indirect approaches, like applying costly optimization procedures or training additional networks [23]. Additionally, GAN networks are notoriously hard to train and are prone to drop modes present in the training data [17, 19]. For these reasons, we focus our attention on VAE-based techniques.

3 VARIATIONAL AUTOENCODER

We first provide a brief review of the Variational Autoencoder (VAE) [10] before extending it to multiple resolutions in Section 4. The VAE consists of (i) a decoder or generator model, $p(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})$, which models the probability distribution of the input data $\mathbf{x}$ conditioned on the low-dimensional representation $\mathbf{z}$ in a latent space and (ii) an encoder or recognition model, $q(\mathbf{z}|\mathbf{x};\boldsymbol{\phi})$, which maps in the other direction. Both models are described using deep neural networks, with parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ respectively.

During training we estimate these parameters such that the marginal likelihood of the training data, $p(\mathbf{x};\boldsymbol{\theta})$, is maximized under a variational Bayesian approximation:

$$\log[p(\mathbf{x};\boldsymbol{\theta})] = D_{KL}[q(\mathbf{z}|\mathbf{x};\boldsymbol{\phi})||p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta})] + L_{VAE}, \quad (1)$$

where the variational lower bound is

$$L_{VAE} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x};\boldsymbol{\phi})}[\log p(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})] - D_{KL}[q(\mathbf{z}|\mathbf{x};\boldsymbol{\phi})||p(\mathbf{z};\boldsymbol{\theta})]. \quad (2)$$

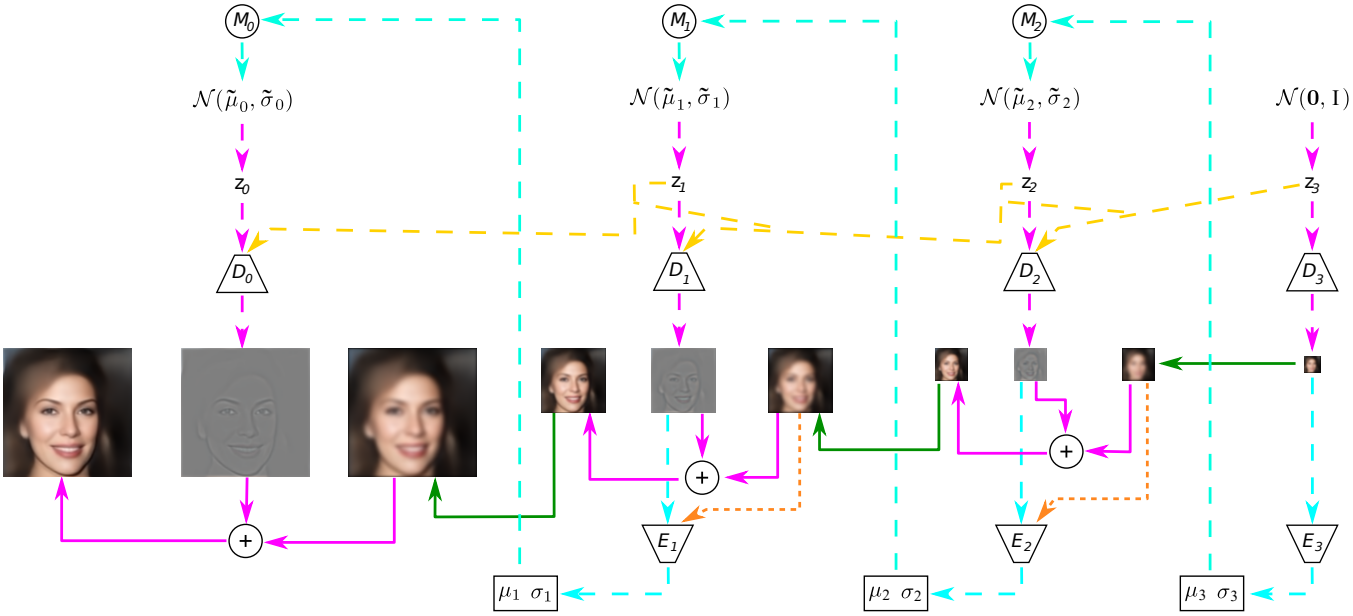

Figure 2: LapCVAE architecture for sample generation. A point in latent space $\mathbf{z}_{k+1}$ is sampled from a normal distribution and decoded to generate a coarse image. The image is encoded (dashed cyan arrows) to compute its particular latent mean $\boldsymbol{\mu}_{k+1}$ and variance σ_{k+1}, and a network M_k is used to estimate the mean $\tilde{\mu}_k$ and variance $\tilde{\sigma}_k$ for the finer level. A new point in latent space $\mathbf{z}_k$ is sampled and the high-frequency image is generated conditioned (dotted yellow arrow) on $\mathbf{z}_{k+1}$. This is added to the upsampled (green arrow) coarse image. The process is repeated until the final result is generated.

In Eq. 1, the left-hand side denotes the log likelihood of the data, and the first term on the right-hand side measures the distance between the approximate encoding distribution and the true posterior. In the variational lower bound, the first term is the reconstruction error (the log probability distribution of the decoder given $q(\mathbf{z}|\mathbf{x};\boldsymbol{\phi})$), and the second term is the divergence between the encoder distribution and a known prior.

The assumption is that the recognition model $q(\mathbf{z}|\mathbf{x};\boldsymbol{\phi})$ will be a good approximation of the intractable posterior $p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta})$, therefore making the intractable non-negative KL divergence in Eq. 1 approach zero. Thus, maximizing the bound will be approximately equivalent to maximizing the marginal likelihood of the data $p(\mathbf{x};\boldsymbol{\theta})$ under the model.

To facilitate computation of the gradients during training, the approximate posterior is defined by a diagonal Gaussian, with mean and variance as a function of the input data

$$q(\mathbf{z}|\mathbf{x};\boldsymbol{\phi}) = \mathcal{N}(\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\sigma}^2(\mathbf{x})\mathbf{I}). \tag{3}$$

The addition of the auxiliary noise vector ϵ is known as the "reparameterization trick" [10], and it allows the sampling of $\mathbf{z}$ to be differentiable

$$\mathbf{z} = \boldsymbol{\mu}(\mathbf{x}) + \boldsymbol{\sigma}(\mathbf{x}) \odot \epsilon, \tag{4}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are functions of $\mathbf{x}$, $\epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ is an auxiliary random noise vector and $\odot$ denotes element wise product.

The choice of prior for $p(\mathbf{z})$ is a Gaussian distribution with zero mean and unit variance

$$p(\mathbf{z};\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0},\mathbf{I}). \tag{5}$$

This straightforward prior is used because it simplifies the KL divergence term in the right-hand side of equation 2 to $\sum_j^J (1+\log(\sigma_j^2) - \mu_j^2 - \sigma_j^2)$, where J is the dimensionality of $\mathbf{z}$.

4 LAPLACIAN VARIATIONAL AUTOENCODER

In this section we present our model, the Laplacian Conditional Variational Autoencoder (LapCVAE). Similarly to the LAPGAN [4] model, the image is generated in a coarse-to-fine manner by treating it as a Laplacian pyramid. At each stage the network only generates details at a specific scale that are added to an upsampled estimate of the image given by the preceeding levels. In this model, the loss function explicitly penalizes errors in generating the high frequency images rather than the overall image reconstruction. The model architecture during generation and training for a pyramid with three levels is shown in Figures 2 and 3.

A Laplacian pyramid is an invertible image decomposition, describing an image as a set of images which encode high-frequency details at different scales. To generate a K level pyramid, a blur and downsample operator $d(\cdot)$ is repeatedly applied to the input image $\mathbf{x}$ to create blurred images $[\mathbf{b}_0, \mathbf{b}_1, \cdots \mathbf{b}_K]$, with $\mathbf{b}_0 = \mathbf{x}$. The image $\mathbf{h}_k$ at level k of the Laplacian pyramid is computed as the difference between the blurred image at level k and an upsampled version of the blurred image at the next level

$$\mathbf{h}_k = \mathbf{b}_k - u(\mathbf{b}_{k+1}), \tag{6}$$

where $u(\cdot)$ is an upsample operator, and the residual is used for the last level, $\mathbf{h}_K = \mathbf{b}_K$. To reconstruct an image $\mathbf{x}$ from its Laplacian

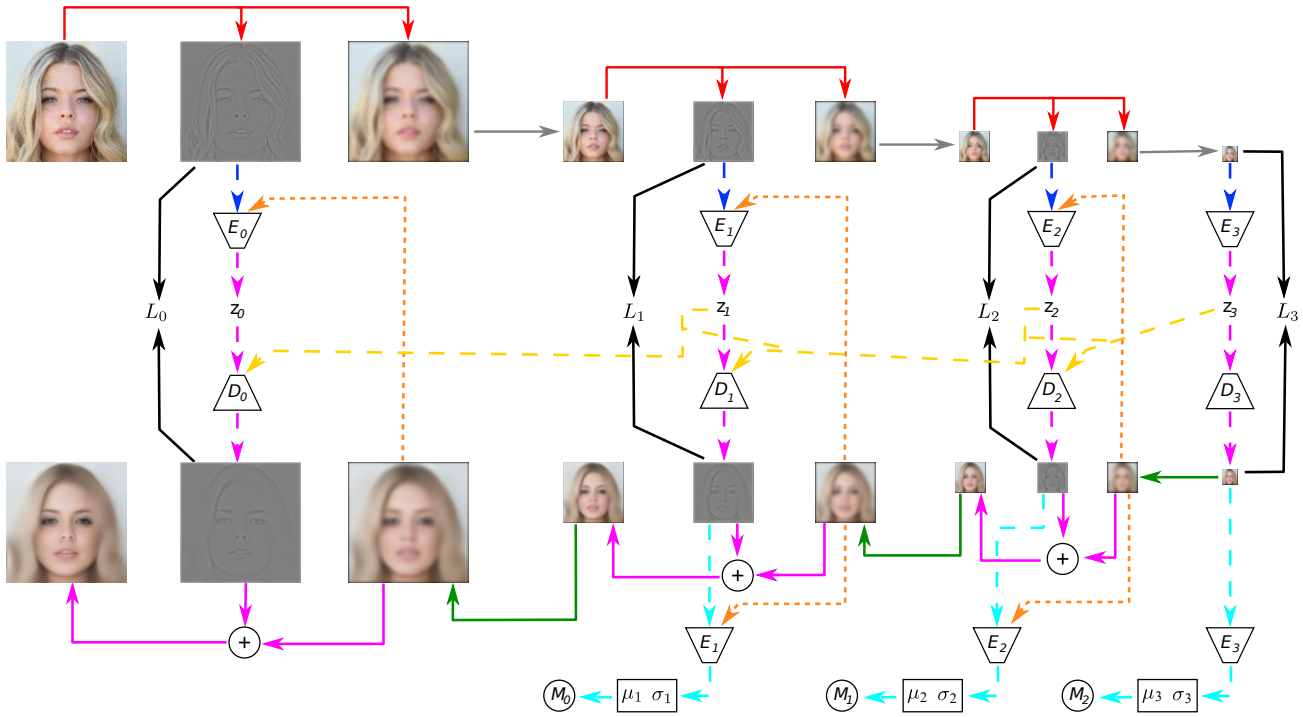

Figure 3: LapCVAE architecture during training. An image is decomposed (red arrows) into its high-frequency and low frequency components. The low-frequency is downsampled (gray arrows). Each VAE learns how to generate the high-frequency ground truth image that receives as input (blue arrows). The encoder also receives as input the composed image generated from the previous levels (dotted orange arrows). The generated image is re-encoded to compute the means and variances for the M_k networks.

pyramid we recursively apply

$$\mathbf{x}_k = \mathbf{h}_k + u(\mathbf{x}_{k+1}), \tag{7}$$

where the recursion starts with $\mathbf{x}_K = \mathbf{h}_K$ and ends with $\mathbf{x} = \mathbf{x}_0$.

In our model, a series of VAEs are used to generate the images in the Laplacian pyramid

$$\mathbf{h}_k \sim p_k(\mathbf{h}_k|\mathbf{z}_k;\boldsymbol{\theta}_k), \tag{8}$$

where $p_k(\cdot)$ is the decoder of the k^{th} VAE, $\mathbf{z}_k$ is an encoding for the high frequency image $\mathbf{h}_k$, and $\boldsymbol{\theta}_k$ denotes the decoder parameters. For each decoder, a corresponding encoder, $\mathbf{z}_k \sim q_k(\mathbf{z}_k|\mathbf{h}_k;\boldsymbol{\phi}_k)$, is defined as part of the VAE.

4.1 Conditioning

To ensure the generated images are coherent across scales, both the encoder and the decoder must be conditioned with the result from the previous level

$$\mathbf{z}_k \sim q_k(\mathbf{z}_k|\mathbf{h}_k, u(\mathbf{x}_{k+1});\boldsymbol{\phi}_k), \tag{9}$$

$$\mathbf{h}_k \sim p_k(\mathbf{h}_k|\mathbf{z}_k, u(\mathbf{x}_{k+1});\boldsymbol{\theta}_k). \tag{10}$$

This formulation conditions the generative process on the previous scale via the upsampled image $u(\mathbf{x}_{k+1})$, but a superior approach for the decoder is to condition the generation on all of the previous latent variables

$$\mathbf{h}_k \sim p_k(\mathbf{h}_k|\mathbf{z}_k, \mathbf{z}_{k+1}, \cdots, \mathbf{z}_K;\boldsymbol{\theta}_k). \tag{11}$$

In our model the encoder only receives the images $\mathbf{h}_k$ and $u(\mathbf{x}_{k+1})$, which are concatenated along the channel dimension, and the resulting tensor is used as input for the k^{th} network, as shown by the orange arrow in Figure 3. The decoder inputs consist of latent vectors, $\mathbf{z}_k, \mathbf{z}_{k+1}, \cdots, \mathbf{z}_K$, which are also concatenated, as shown by the yellow arrow in Figure 3.

4.2 Loss function

The loss function for each VAE in the pyramid is defined as:

$$L = \mathbb{E}_{q_k(\mathbf{z}_k|\mathbf{h}_k, u(\mathbf{x}_k);\boldsymbol{\phi}_k)}[\log p_k(\mathbf{h}_k|\mathbf{z}_k, \mathbf{z}_{k+1}, \cdots, \mathbf{z}_K;\boldsymbol{\theta}_k)] - \lambda_k D_{KL}\left[q_k(\mathbf{z}_k|\mathbf{h}_k, u(\mathbf{x}_{k+1});\boldsymbol{\phi}_k)||p(\mathbf{z}_k;\boldsymbol{\theta}_k)\right], \tag{12}$$

where λ_k is a scalar hyper-parameter that serves as a tradeoff between the reconstruction loss and the distance from the encoding prior. For the reconstruction loss we measure the error between the input Laplacian pyramid images, and the ones generated by the network, as shown in black arrows in Figure 3.

4.3 Latent space prior distribution

Hoffman and Johnson [9] discussed the need for better priors in VAE networks. Taking advangate of our pyramid architecture, we propose a novel Gaussian prior. In similar spirit as Gulrajani *et al.* [8], our prior encourages the latent space of a given level to be a transformation of the coarser level's latent distribution. The prior distribution

for our model is defined as

$$p(\mathbf{z}_k;\boldsymbol{\theta}_k) = \mathcal{N}\big(R_k(\boldsymbol{\mu}_{k+1};\boldsymbol{\xi}_k), S_k(\boldsymbol{\sigma}_{k+1};\boldsymbol{\omega}_k)\big), \tag{13}$$

where $\boldsymbol{\mu}_{k+1}$ and $\boldsymbol{\sigma}_{k+1}$ are the mean and standard deviation of the latent point in the coarser level as in eq. 4, $R_k(\cdot)$ and $S_k(\cdot)$ are a family of transformation functions parametrized by $\boldsymbol{\xi}_k$ and $\boldsymbol{\omega}_k$.

For the coarsest level in the pyramid the standard VAE prior defined in equation 5 is used. For simplicity, we will use $M_k(\cdot)$ to denote the pair of functions $R_k(\cdot)$ and $S_k(\cdot)$.

To ensure equivalent distributions at train time (where the original image is available) and test time (where it is not), $\boldsymbol{\mu}_{k+1}$ and $\boldsymbol{\sigma}_{k+1}$ are given by encoding the reconstructed image from $p_{k+1}(\cdot)$ with $q_{k+1}(\cdot)$.

Using the parameters defined by M_k gives more flexibility to the latent space distribution in the finer levels of the pyramid. As the means and variances of the Gaussian distribution are image dependent, the distribution of the latent space can be regarded as an mixture of Gaussians.

We approximate the M_k functions using shallow multilayer perceptron networks. The parameters $\boldsymbol{\xi}_k$ and $\boldsymbol{\omega}_k$ can be jointly learned with minimal overhead during training, by minimizing equation 12.

5 RESULTS

We evaluated two versions of our model on the CelebA [14] dataset, using 64×64 and 128×128 images taken from the aligned and cropped version of the dataset. The dataset consists of 202,599 images of faces, we use 80% of the images for training and 20% for testing. We trained two models, with four Laplacian levels for the 64×64 and with five for the 128×128, where each downsampling operator halves the image size, i.e. for the 64×64 version the input size for each VAE is 64×64, 32×32, 16×16 and 8×8 pixels.

The mean squared error (MSE) between pixels was used as the reconstruction cost in the loss function. The size of the latent space for each level of the pyramid is set to 64 for the 64×64 network, and $\mathbf{n_z} = [128,128,64,64,64]$ for the larger model. I.e. the small model has a combined dimensionality for the latent space of 256, and the large model of 448.

The value of λ_k, the weight for the KL divergence term for M_k, acts as a trade-off between the reconstruction and the sampling quality. We performed a greedy search starting at the coarsest level of the pyramid for the value that would yield reasonable samples and reconstructions. The results shown in this section use $\boldsymbol{\lambda} = [10,0.1,0.01]$ for the 64×64 model, and $\boldsymbol{\lambda} = [100,100,0.1,0.01]$ for the 128×128.

The model was implemented in Tensorflow [1] and trained on a single Nvidia Titan X. The whole pyramid model takes approximately 8 hours to train for the 64×64 model and 14 for the 128×128. For more details on the network architecture please refer to the supplementary material.

5.1 Comparison with related work

We compare our model in terms of image quality and complexity to VAE [10], VAE/GAN [13] and PixelVAE [8]. The PixelVAE model extends VAEs by adding PixelCNN layers after the decoder. We use the two level version of the model. VAE/GAN is a hybrid of VAE and GAN, which consists of an encoder, decoder and a discriminator. The authors replace the L2 reconstruction loss, with a loss in feature space and an adversarial loss. For PixelVAE and VAE/GAN we trained the models using the code provided by the authors without any modifications to the architecture. The VAE were trained with the same latent space dimensionality and similar number of parameters as our models.

Figure 4: Comparison of image reconstruction results for different AutoEncoder architectures on the test data. VAE/GAN and PixelVAE add additional details in the images that do not necessarily correspond to the input. Our architecture produces sharper images than VAE. This is more noticeable in the last two rows that show results for the 128×128 models.

5.1.1 Reconstructions. Examples of reconstructions are shown in Figure 4. VAE reconstructions are blurrier than the other methods, but are able to capture the main features in the faces. PixelVAE and VAE/GAN produce sharper results, however the added detail does not necessarily correspond with the input images. For PixelVAE this is due to the local nature of the autoregressive PixelCNN layers, and in VAE/GAN due to the inclusion of the adversarial loss. Our model is able to produce sharper images in comparison to VAE.

Model	Reconstruction error	Network parameters
VAE [10]	22.78±4.64*	6.58×10^7
VAE/GAN [13]	30.49±7.32*	4.69×10^7
LapCVAE Without-M	22.19±4.80	6.54×10^7
LapCVAE	**20.60±4.81**	6.55×10^7
VAE [10] 128×128	20.75±4.40*	1.68×10^8
LapCVAE 128×128	**20.61±5.15**	1.66×10^8

Table 1: Quantitative comparison of model complexity and reconstruction error, measured as the mean of the square root of the MSE with standard deviation. Models trained with this error metric are denoted by a * next to their error value.

Quantitative results are presented in Table 1, measured as the mean and the standard deviation of the square root of the MSE over all images, with pixels in the range [0,255]. The network complexity is measured in floating point operations. We do not provide a mean reconstruction error for PixelVAE, given how costly it is to generate images with this method, around 40 minutes for a batch of 64 images. LapCVAE both with and without the M functions are able to reconstruct images with lower error than VAE and VAE/GAN.

At the 64×64 resolution the effects of using the Laplacian architecture instead of a VAE are rather subtle. However, the advantages of using our model becomes more evident as the resolution increases. The PixelVAE and VAE/GAN were designed by the authors to only handle 64×64 images, and due to GPU RAM constrains and need of parameter tuning we were unable to extend them to 128×128. Our model is less effective at representing the colors in the images, yet it is able to better capture sharp features and small details. For example, note in the first row of Figure 4 how the gaze and mouth are better reproduced by our model.

5.1.2 Human evaluation of reconstructions. To further demonstrate the advantages of our method, we perform a small user study to evaluate the perceptual quality of the reconstructed images. We compare 128×128 reconstructions of VAE and LapCVAE. We performed two different experiments where four participants evaluated 500 images each. In the first experiment, the participants were presented with the reconstructions of the same image by both methods and had to choose the one with the highest quality. In the second experiment, the participants were presented with the reconstructions together with the original image and were asked to choose the reconstruction that better matched the original. For both experiments the order in which the two reconstructions were shown was randomized.

For both experiments our reconstructions were preferred over the competing method by a large margin, as shown in Table 2. These results seem to indicate that the reconstructions obtained with our method, despite being similar in terms of reconstruction error (see Table 1), are superior to the reconstructions obtained with VAE. The added detail and sharpness visible in our reconstructions is important in human perception of image quality.

Model	Human preference %	
	Without original	With original
VAE [10] 128×128	15.61±8.14	26.30±7.35
LapCVAE 128×128	**84.39±8.14**	**73.70±7.35**

Table 2: Human evaluation of pairs of reconstructions from VAE and our model, with and without the input image. Images generated with our model are preferred over the competing method.

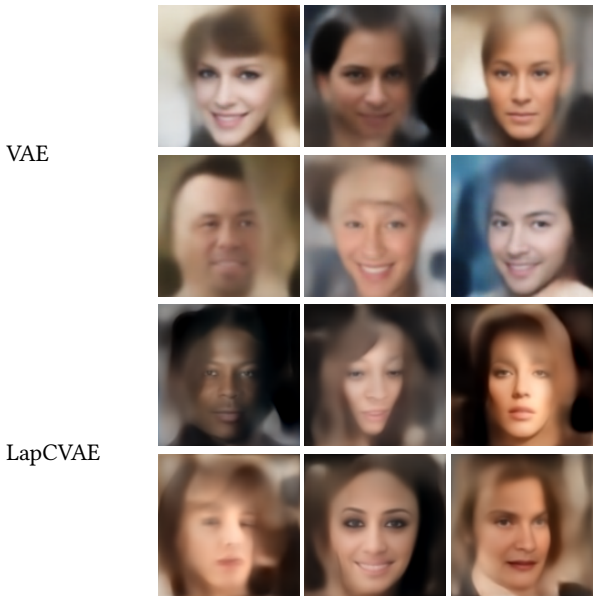

Figure 5: The samples from our model and VAE on the 128×128 architecture. The samples from LapCVAE appear to be as varied and with similar quality as the ones from VAE.

5.1.3 Samples. Some samples from our model are shown in Figure 5. The main features of the face are seen clearly, yet the background and hair are noticeably blurry. The samples present some variety of pose and style, and they appear to be similar in terms of quality to VAE samples.

In order to test that the network did not overfit, the nearest neighbours in the training set for a number of samples are shown in Figure 6. The similarity between the samples and the training data is measured in pixel space using the mean squared error. The samples are distinct from their neighbors, which demonstrates that the model is not just memorizing the training images.

Our pyramid architecture explicitly encodes coarse to fine information in the latent representation learned by the model. This effect is shown in Figure 7, where the z for the coarser levels are fixed, and a number of images are produced by sampling the remaining levels. Less variation in the samples is observed as more levels are fixed.

During training the loss function contains two terms, the reconstruction and the KL divergence, which are potentially conflicting.

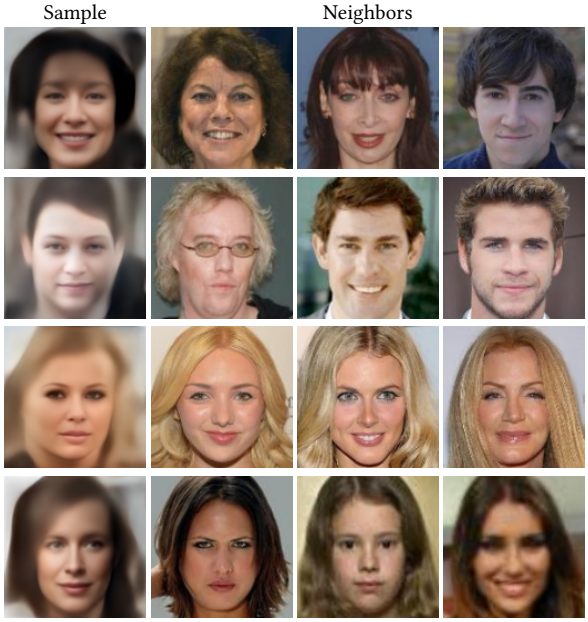

Figure 6: Samples from our 128×128 method with nearest neighbors in the training data, which show that the model is not simply memorizing the images in the dataset.

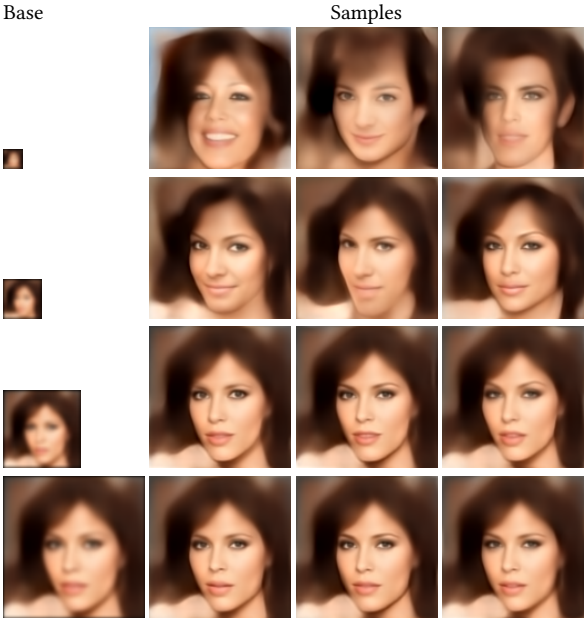

Figure 7: Sampling with $z_{k \cdots K}$ fixed at different levels of the pyramid with the 128×128 model. The *Base* columns contain the blurred and upsampled image generated by the fixed levels, and the remaining images are generated by sampling different latent vectors at the rest of the levels $z_{0 \cdots k-1}$. As expected the samples are consistent with the coarse base and the variety decreases as more levels are fixed.

After convergence, the sample quality can be improved by retraining the M networks (see Section 4.3), while keeping the rest of the model fixed. A comparison of samples generated before and after retraining M is shown in Figure 8. Some improvements, such as extra sharpness or smoother color transitions can be noticed in the retrained examples. The samples generated for any other figure in the paper were generated without retraining M.

5.2 Image editing

Our method allows partial sampling, where the coarser levels of the pyramid are kept fixed and the finer levels are sampled, as shown in Figure 7. This enables novel image editing applications that are not possible with VAE. Image editing examples using our 128×128 model are shown in Figures 9 and 1.

For an input image, the user can select a region to be modified (Figure 9) or edit the coarse level image directly (Figure 1). From a coarse input, a number of novel images are sampled from the network, providing a large variety of compatible results. The final image can the be composited back [16] easily as the generated images are, by construction. compatible with the input, *i.e.*the patch is already mostly aligned and it has similar colors.

For example, in Figure 1 right, our method is used to remove the sunglasses from input image . On the coarse level, the area around the eyes is painted with skin tone, and several samples of faces without sunglasses are generated from it. The region of interested is then composited with the reconstructed image.

Figure 8: Sample comparison with our 128×128 model and retraining M after convergence. After retraining the generated images are sharper and contain less artifacts.

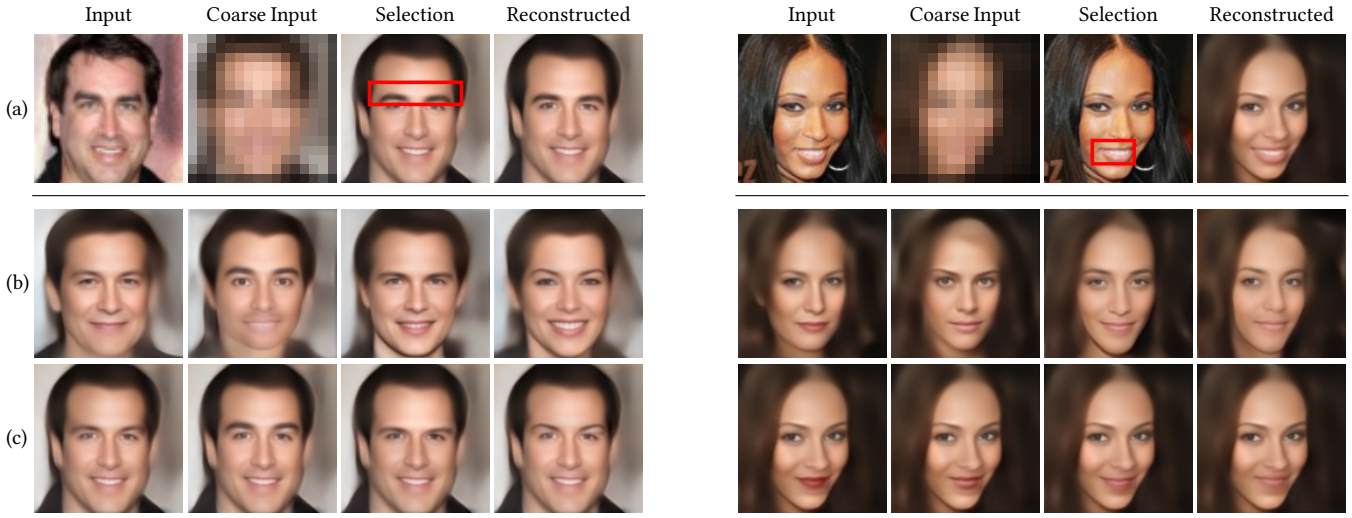

Figure 9: Image editing example. (a) An input image is decomposed using a Laplacian pyramid. The user selects an area to be edited. The input image is projected to a low-dimensional representation and reconstructed from it. (b) Several samples are generated conditioned on the coarse input creating interesting variations. (c) The selected regions from the samples are composed with the reconstructed image.

6 CONCLUSIONS AND FUTURE WORK

In this paper we presented Laplacian Pyramid of Conditional Variational Autoencoder (LapCVAE), a conditional multi-scale extension of Variational Autoencoder (VAE) models. Our generative network is able to produce reconstructions and samples, which are sharper than VAE results and at higher resolution than alternative methods. It can be trained in a greedy fashion and it provides more flexibility, as it allows partial sampling of only some of the levels of the pyramid.

In terms of limitations, the images produced by our network lack some of the overall colors that a VAE of similar complexity is able to capture. Moreover, the total training time is doubled if the greedy training procedure is used, and the image generation is less efficient as there are several levels to sample from.

There are a number of opportunities for future work. Our method still fails to capture local high frequency detail. To overcome this limitation, a PixelCNN network could be added after each VAE in the pyramid, in a similar fashion as Gulrajani *et al.* [8]. A different approach to improve the quality of the generated images is to use a perceptual or adversarial loss for the reconstruction error, analogous to the work in Larsen *et al.* [13]. The effects of using our latent space loss could benefit from a formal mathematical study.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). Software available from tensorflow.org.

[2] Andrew Brock, Theodore Lim, JM Ritchie, and Nick Weston. 2016. Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093* (2016).

[3] Garrison W Cottrell, Paul Munro, and David Zipser. 1987. Learning internal representations from gray-scale images: An example of extensional programming. In *Conference of the Cognitive Science Society*.

[4] Emily L Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. 2015. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. In *Advances in Neural Information Processing Systems*. 1486–1494.

[5] Jon Gauthier. 2014. Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester* 2014 (2014), 5.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. 2672–2680.

[8] Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. 2017. PixelVAE: A Latent Variable Model for Natural Images. In *International Conference on Learning Representations (ICLR)*.

[9] Matthew D Hoffman and Matthew J Johnson. 2016. ELBO surgery: yet another way to carve up the variational evidence lower bound. In *NIPS 2016 Workshop on Advances in Approximate Bayesian Inference*.

[10] Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. *International Conference on Learning Representations* (2014).

[11] Alexander Kolesnikov and Christoph H Lampert. 2016. Deep Probabilistic Modeling of Natural Images using a Pyramid Decomposition. *arXiv preprint arXiv:1612.08185* (2016).

[12] A. Lamb, V. Dumoulin, and A. Courville. 2016. Discriminative Regularization for Generative Models. *arXiv preprint arXiv:1612.03220* (2016).

[13] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, and Ole Winther. 2016. Autoencoding beyond pixels using a learned similarity metric. In *Proceedings of The 33rd International Conference on Machine Learning*, Vol. 48. JMLR, 1558–1566.

[14] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

[15] Aaron Van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. 2016. Pixel Recurrent Neural Networks. In *Proceedings of The 33rd International Conference on Machine Learning*, Vol. 48. 1747–1756.

[16] Patrick Pérez, Michel Gangnet, and Andrew Blake. 2003. Poisson Image Editing. *ACM Trans. Graph.* 22, 3 (July 2003), 313–318.

[17] Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations (ICLR)*.

[18] Scott Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. 2016. Learning What and Where to Draw. In *NIPS*.

[19] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. 2016. Improved Techniques for Training GANs. In *Advances*

in Neural Information Processing Systems 29. 2234–2242.

[20] Paul Upchurch, Jacob Gardner, Kavita Bala, Robert Pless, Noah Snavely, and Kilian Weinberger. 2016. Deep Feature Interpolation for Image Content Changes. *arXiv preprint arXiv:1611.05507* (2016).

[21] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2Image: Conditional Image Generation from Visual Attributes. *Proceedings of European Conference on Computer Vision (ECCV)* (2016).

[22] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. 2016. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. *arXiv preprint arXiv:1612.03242* (2016).

[23] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. 2016. Generative Visual Manipulation on the Natural Image Manifold. In *Proceedings of European Conference on Computer Vision (ECCV)*.