Abstract
We study probabilistic models of natural images and extend the autoregressive family of PixelCNN models by incorporating latent variables. Subsequently, we describe two new generative image models that exploit different image transformations as latent variables: a quantized grayscale view of the image or a multi-resolution image pyramid. The proposed models tackle two known shortcomings of existing PixelCNN models: 1) their tendency to focus on low-level image details, while largely ignoring high-level image information, such as object shapes, and 2) their computationally costly procedure for image sampling. We experimentally demonstrate benefits of our LatentPixelCNN models, in particular showing that they produce much more realistically looking image samples than previous state-of-the-art probabilistic models.

1. Introduction
Modeling the distribution of intensities in natural images is a fundamental statistical problem. Natural images are the main input to visual systems and, thus, understanding their structure is important for building strong and accurate vision systems. Image modeling is also useful for a wide variety of key computer vision tasks, such as visual representation learning, image inpainting, deblurring, super-resolution, image compression, and others.

Natural image modeling is known to be a difficult task. Because the distribution over natural images is highly complex, developing models that are both accurate and computationally tractable is very challenging. Until recently, most existing models were restricted to modeling very small image patches, no bigger than, e.g., 9x9 pixels. Recently, however, deep convolutional neural networks (CNNs) have triggered noticeable advances in probabilistic image modeling. Out of these, PixelCNN-type models (van den Oord et al., 2016a;b; Salimans et al., 2017), have shown to deliver the best performance, while at the same time staying computationally tractable. However, PixelCNNS also have noticeable shortcomings: unless conditioned on external input, the samples they produce rarely reflect global structure of complex natural images, see (van den Oord et al., 2016b; Salimans et al., 2017). This raises concerns that current PixelCNN architectures might also be more limited than originally presumed. Moreover, PixelCNN’s image sampling procedure is relatively slow in practice, as it requires to invoke a very deep neural network for every single image pixel that is to be generated.

In this work we present LatentPixelCNN, a new technique for probabilistic image modeling that overcome several of the aforementioned shortcomings. The main idea is to augment PixelCNN with appropriate latent variables in order to isolate and overcome these drawbacks. This step, at the same time, provides us with important insights into the task of modeling natural image.

Besides the above insight, we make two main technical contributions in this paper. First, we show that uncertainty in low-level image details, such as texture patterns, dominates the objective of ordinary probabilistic PixelCNN models and, thus, the model has little incentive to capture visually essential high-level image information, such as object shapes. We tackle this issue by deriving a Grayscale LatentPixelCNN that effectively decouples the tasks of modeling low and high-level image details. This results in image samples of substantially improved visual quality. Second, we show that the sampling speed of PixelCNN models can be substantially accelerated. We accomplish this by deriving a Pyramid LatentPixelCNN that decomposes the modeling of the image pixel probabilities into a series of much simpler steps. Employing a much lighter-weight PixelCNN architecture for each of them, globally coherent high-resolution samples can be obtained at strongly reduced computational cost.

2. Related Work
Probabilistic image modeling is a research area of long tradition that has attracted interest from many different disciplines (Ruderman & Bialek, 1994; Olshausen & Field, 1996; Hyvärinen et al., 2009). However, because of the
difficulty of the problem, until recently all existing models were restricted to small image patches, typically between 3x3 and 9x9 pixels, and reflected only low-order statistics, such as edge frequency and orientation (Zhu et al., 1998; Roth & Black, 2005; Carlsson et al., 2008; Zoran & Weiss, 2011). Utilized as prior probabilities in combination with other probabilistic models, such as Markov random fields (Geman & Geman, 1984), these models proved to be useful for low-level imaging tasks, such as image denoising and inpainting. Their expressive power is limited, though, as one can see from the fact that samples from the modeled distribution do not resemble natural images, but rather structured noise.

This situation changed with the development of probabilistic image models based on deep neural networks, in particular variational auto-encoders (VAEs) and PixelCNNs. In this paper, we concentrate on the latter. The PixelCNN family of models (van den Oord et al., 2016a;b; Salimans et al., 2017) factorizes the distribution of a natural image using the elementary chain rule over pixels. The factors are modeled as deep convolutional neural networks with shared parameters and trained by maximum likelihood estimation.

VAEs (Kingma & Welling, 2014) offer an alternative approach to probabilistic image modeling. They rely on a variational inequality to bound the intractable true likelihood of an image by a tractable factorized approximation. VAEs are efficient to evaluate, but so far, produce results slightly worse state-of-the-art PixelCNNs, both the likelihood scores and sampled images. Similar to Latent-PixelCNNs, recent advances (Gregor et al., 2015; 2016; Bachman, 2016; Kingma et al., 2016; Gulrajani et al., 2017; Chen et al., 2017) in VAEs literature exploit modifications of model structure, including usage of latent variables and autoregression principle, though these techniques remain technically and conceptually different from PixelCNNs.

Specifically for the task of producing images and other complex high-dimensional objects, generative adversarial networks (GANs) (Goodfellow et al., 2014) have recently gained popularity. In contrast to PixelCNNs and VAEs, GANs are not probabilistic models but feed-forward networks that are directly trained to produce naturally looking images from random inputs. As such, they act more as black boxes and do not explicitly provide insights in the statistical structure of natural images. A drawback of GAN models is that they have a generally unstable training procedure, associated with search of a Nash equilibrium between two competing network players, and they can suffer from various technical problems, such as mode collapse or vanishing gradients. In order to make GANs work in practice, researchers resort to multiple non-trivial heuristics (Salimans et al., 2016). This strongly contrasts with probabilistic autoregressive models, such as PixelCNNs, which rely on well understood likelihood maximization for training and do not suffer from mode collapse problems.

Despite having fundamental differences on the technical level, PixelCNNs, VAEs and GANs may also benefit each other by sharing ideas. In particular, our work is related to the line of work on GANs with controllable image structure (Reed et al., 2016; Wang & Gupta, 2016). A crucial difference of our work is, however, that our models do not require external supervision and, thus, remain purely unsupervised. Another notable paper in the context of this work introduces Laplacian GANs (Denton et al., 2015), with which we share the similar idea of using multi-scale decomposition for image generation. Similar constructions were suggested in (van den Oord et al., 2016b) in the context of recurrent networks and (Dahl et al., 2017) for the problem of super-resolution. We differ from those in many technical details and, most important, our multi-scale model is specifically designed not only to improve sample quality, but also to speed up the sampling process by relying on very light-weight PixelCNN modules.

3. LatentPixelCNN

In this section we remind the reader of the technical background and formally introduce LatentPixelCNNs. We then present two instances that provide interesting insights into the natural image modeling task and lead to improved quality of sampled images and accelerated sampling procedure. Finally, we conclude the section with implementation and training details.

We define an image \( X = (x_1, x_2, \ldots, x_n) \) as a collection of \( n \) random variables associated with some unknown probability measure \( p(X) \). Each random variable represents a 3-channel pixel value in the RGB format, where each channel takes a discrete value from the set \{0, 1, 2, \ldots, 255\}. The pixels are ordered according to a raster scan order: from left to right and from top to bottom. Given a dataset \( D \) of \( N \) images, our main goal is to estimate the unknown probability measure \( p(X) \) from \( D \).

Recall that PixelCNN is a family of models (van den Oord et al., 2016a;b; Salimans et al., 2017) that factorizes the distribution of natural images using the basic chain rule:

\[
p(X) = \prod_{j=1}^{n} p(x_j | x_1, \ldots, x_{j-1}) . \tag{1}\]

The key idea of LatentPixelCNNs is to introduce an additional latent variable, \( \tilde{X} \), into the image modeling process. Formally, a LatentPixelCNN is a probabilistic model of the joint distribution of \( X \) and \( \tilde{X} \), factorized as

\[
p(X, \tilde{X}) = p_{\theta}(\tilde{X}) p_{\phi}(X | \tilde{X}), \tag{2}\]
where \([\theta, \tilde{\theta}]\) are parameters of the probabilistic models. The conditional probability distribution, \(p_\theta(X|\tilde{X})\), is modeled using the PixelCNN model (van den Oord et al., 2016b), including the recent improvements suggested in (Salimans et al., 2017):

\[
p_\theta(X|\tilde{X}) = \prod_{j=1}^{n} p_\theta(x_j|x_1,\ldots,x_{j-1}; f_w(\tilde{x})) ,
\]

where \(f_w\) is an embedding function parametrized by \(w\).

Like other PixelCNN-based models, our model can be used for drawing samples. For a fixed \(\tilde{X}\), one follows the ordinary pixel-wise PixelCNNS sampling strategy. Otherwise, one first samples \(X\) from \(p_\theta(\tilde{X})\) and then samples \(X\) from \(p_\theta(X|\tilde{X})\) as described before. This, and the fact that we want to learn not only \(p_\theta(X|\tilde{X})\) but also \(p_\theta(\tilde{X})\) from data, suggest a specific form of the latent variables.

Specifically, in this work we concentrate on latent variables that 1) are a form of images themselves such that we can model \(p_\theta(\tilde{X})\) by a PixelCNN-type model and avoid the need to mix different probabilistic modeling frameworks, and 2) for which \(\tilde{X}\) is approximately computable by a known deterministic function \(\psi : X \rightarrow \tilde{X}\). This choice has the useful consequence that the PixelCNN’s loss function (the negative data log-likelihood) measures the amount of uncertainty of the color value of each pixel, conditioned on the previous ones. This quantity is dominated by image region with hard-to-predict low-level cues, such as texture patterns, where each pixel exhibits a large uncertainty, even if its neighbors are known. As a consequence, the probabilistic model is encouraged to represent such textures well, while visually more essential image details, such as object shapes, are neglected. We provide quantitative evidence for this claim in our experimental section.

3.1. Grayscale LatentPixelCNN

Despite great success of PixelCNN-type models, they are still far away from producing plausible samples of complex natural scenes. A visual inspection of samples produced by the current state-of-the-art models reveals that they typically match low-level image details well, but fail at capturing global image structure, such as object shapes, see Figure 1 for an illustration.

We conjecture that a major reason for this is that the PixelCNN training objective provides too little incentive for the model to actually capture high-level image structure. Concretely, the PixelCNN’s loss function (the negative data log-likelihood) measures the amount of uncertainty of the color value of each pixel, conditioned on the previous ones. This quantity is dominated by image region with hard-to-predict low-level cues, such as texture patterns, where each pixel exhibits a large uncertainty, even if its neighbors are known. As a consequence, the probabilistic model is encouraged to represent such textures well, while visually more essential image details, such as object shapes, are neglected. We provide quantitative evidence for this claim in our experimental section.

In order to tackle the aforementioned shortcoming we employ an instance of the LatentPixelCNN, in which the latent variable \(\tilde{X}\) is a 4-bit per pixel quantized grayscale version of the original 24-bits color image \(X\). In combination with the factorization (2), this choice of latent variable decouples the modeling of low and high-level image details: the distribution \(p_\theta(\tilde{X})\) (the quantized grayscale image) contains most information about global image properties, reflecting present objects and their shapes. The distribution \(p_\theta(X|\tilde{X})\) models missing color and texture details. In Section 4 we highlight the quantitative and qualitative effects of augmenting PixelCNN with this choice of latent variable.
3.2. Pyramid LatentPixelCNN

In this section we address two further shortcomings of existing PixelCNN models. First, the strong asymmetry of the factors in Equation (1): the top left pixel of an image is modeled unconditionally, i.e., without any available information, while the bottom right pixel has access to the information of, potentially, all other pixel values. Nevertheless, the same network is evaluated to model either of them (as well as all others pixels in between). We conjecture that it would be beneficial if pixels were generated with a less asymmetric usage of the image information.

Second, PixelCNNs have high computational cost for sampling images due to the recurrent nature of the procedure: for each generated pixel, the PixelCNN must invoke a convolutional neural network that is very deep, often in the order of one hundred convolutional layers. Note, that, in principle, at the sampling phase PixelCNN allows to cache intermediate values across consecutive PixelCNN invocations and, thus, save considerable computational effort. However, as reported in (Ramachandran et al., 2017), caching delivers minor sampling speed gain, when only a few or a single image is generated at once, which is a common scenario in real-life applications.

In order to alleviate the aforementioned drawbacks we propose a LatentPixelCNN in which the latent variable \( \hat{X} \) corresponds to a twice lower resolution view of \( X \), thereby decoupling the full image model into a pair of simpler problems: creating a lower resolution image, and upsampling a low-res into a high-res image. The upsampling step has strongly reduced model asymmetry, because all pixel on the high scale have equal access to all information from the lower scale. Also, by explicitly modeling the low-resolution image view we make it easier for the model to capture long-range image correlations, simply because the “long range” now stretches over fewer pixels.

Since the proposed latent variable is an ordinary image, we can recursively apply the same decomposition to model the latent variable itself. For example, if we apply decomposition recursively 4 times for modeling 128x128 images, it will result in a model, which first generates an image on 8x8 resolution, and then upscales it 4 times by a factor 2. We call the resulting model Pyramid LatentPixelCNN, as it resembles image pyramid decomposition.

Note, that Pyramid LatentPixelCNN breaks the image model into a series of simpler sub-models. Each sub-model is determined by the corresponding conditional distribution \( p_{x_i}(X \mid \hat{X}) \) and the embedding \( f_w(\hat{X}) \) (or just by \( p_{\theta}(\hat{X}) \) for the lowest resolution image). These conditional distributions model relatively simple task of producing an image, given an embedding of the slightly lower resolution view of this image. Thus, we hypothesize that with an appropriate embedding function (potentially modeled by a very deep network), the conditional distributions can be reliably modeled using a very light-weight network. We then expect a significant sampling speed acceleration, because the major part of computational burden is redistributed to the embedding functions \( f_w(\hat{X}) \), which needs to be computed only once per pyramid layer, not once for each pixel.

We experimentally quantify the modeling performance and sampling speed of the proposed multi-scale image model later in Section 4.

3.3. Details of model parameterization

The LatentPixelCNN model is fully defined by factors in (2), each of which is realized by a network with the PixelCNN++ architecture. This is described in details in (Salimans et al., 2017).

The output of the PixelCNN++ is a 10-component mixture of three-dimensional logistic distributions that is followed by a discretized likelihood function (Kingma et al., 2016; Salimans et al., 2017). The only exception is the model for 4-bit quantized grayscale latent variable \( \hat{X} \), where output is a vector of 16 probabilities for every possible grayscale value, followed by the standard cross-entropy loss.

The conditional PixelCNN model, \( p(X \mid \hat{X}) \), depends on latent variable \( \hat{X} \) in a fashion similar to (van den Oord et al., 2016a; Salimans et al., 2017). It can be summarized in two steps as follows: compute an embedding of \( \hat{X} \) using a convolutional network \( f_w(\hat{X}) \), and bias the convolutions of every residual block by adding the computed embedding. We choose the architecture for modeling the embedding function, \( f_w(\hat{X}) \), to be almost identical to the architecture of PixelCNN++. The main difference is that we use only one flow of residual blocks and do not shift the convolutional layers outputs, because there is no need to impose sequential dependency structure on the pixel level.

For numeric optimization, we use Adam (Kingma & Ba, 2014), a variant of stochastic gradient optimization. During training we use dropout with 0.5 rate, in a way that suggested in (Salimans et al., 2017). No explicit regularization is used.

Further implementation details, such as number of layers are specified later in Section 4.

We have implemented LatentPixelCNNs using tensorflow and will publish the code soon.

4. Experiments

In this section we experimentally evaluate the proposed LatentPixelCNNs on real image modeling task. We report quantitative and qualitative results and discuss new insights
resulting from the latent variable decompositions.

4.1. Grayscale LatentPixelCNN.

Experimental setup. We evaluate the modeling performance of the Grayscale LatentPixelCNN on the CIFAR-10 dataset (Krizhevsky & Hinton, 2009). It consists of 60,000 natural images of size 32 × 32 belonging to 10 semantic categories. The dataset is split into two parts: a training set with 50,000 images and a test set with 10,000 images.

For setting up the architectures for modeling distributions \( p_\theta(\hat{X}) \), \( p_\theta(X|\hat{X}) \) and embedding \( f_w(X) \) we use the same hyperparameters as in (Salimans et al., 2017). The only exception is that for parameterizing \( p(X|\hat{X}) \) and \( f_w(\hat{X}) \) we use 24 residual blocks instead of 36.

In the Adam optimizer we use an initial learning rate of 0.001, a batch size of 64 images and exponential learning rate decay of 0.99999 that is applied after each iteration.

Modeling performance. The Grayscale LatentPixelCNN achieves an upper bound on the negative log-likelihood score of 2.98 bits-per-dimension. This is on par with current state-of-the art models, see Table 1. Note, that since we measure an upper bound, the actual model performance might be slightly better. However, in light of our and other experiments, we believe small differences in this score to be of minor importance, as the log-likelihood does not seem to correlate well with visual quality in this regime.

In Figure 2 we present random samples produced by the Grayscale LatentPixelCNN model, demonstrating grayscale samples from \( p_\theta(\hat{X}) \) and embedding \( f_w(X) \). We observe that the produced samples are highly diverse, and, unlike samples from previously proposed probabilistic autoregressive models, often exhibit a strongly coherent global structure, resembling highly complex objects, such as cars, dogs, horses, etc.

Given the high quality of the samples, one might be worried if possibly the grayscale model, \( p_\theta(\hat{X}) \), had overfit the training data. We observe that training and test loss of \( p_\theta(\hat{X}) \) are very close to each other, namely 0.442 and 0.459 of bits-per-dimension, which speaks against significant overfitting. Note also, that it is not clear if an overfit model would produce good samples as all. For instance, as reported in (Salimans et al., 2017), severe overfitting of the PixelCNN++ model does not lead to the high perceptual quality of sampled images.

Discussion. By explicitly emphasizing the modeling of high-level image structures in the Grayscale LatentPixelCNN, we achieve significantly better visual quality of the produced samples. This suggests that the family of PixelCNN models is sufficiently powerful to capture the structure of complex natural images.

Additionally, the Grayscale LatentPixelCNN offers interesting insights into the image modeling task. As the objective we minimize for training is a sum of two scores, we can individually examine the performance of latent variable model \( p(\hat{X}) \) and the conditional model \( p(X|\hat{X}) \). The trained conditional model achieves a negative log-likelihood score of 2.52 bits-per-dimension, which is significantly lower than the overall log-likelihood. We take this as indication that if low-level and high-level image details are modeled by one model, then at training time low-level image uncertainty will dominate the training objective. We believe that this is why previous PixelCNN models failed to produce globally coherent samples of natural images. Importantly, this problem does not appear in Grayscale LatentPixelCNN, because global and local image models do not share parameters and, thus, do not interfere with each other at training phase.

An alternative explanation for the differences in log-likelihood scores would be that PixelCNN-type models are actually not very good at modeling low-level image input. Additional experiments that we performed show that this is not the case: we applied the learned conditional model \( p(X|\hat{X}) \) to 4-bit grayscale images obtained by quantizing real images of the CIFAR-10 test set. Figure 3 compares the resulting colorized samples with the corresponding original images, showing that the samples produced by our conditional model are of visual quality comparable to the original images. This suggests that in order to produce even better image samples, mainly improved models for \( p(\hat{X}) \) are required.

| Model                          | Bits per dim. |
|-------------------------------|---------------|
| Deep Diffusion (Sohl-Dickstein et al., 2015) | ≤ 5.40        |
| NICE (Dinh et al., 2014)      | 4.48          |
| DRAW (Gregor et al., 2015)    | ≤ 4.13        |
| Deep GMMS (van den Oord & Schrauwen, 2014) | 4.00          |
| Conv Draw (Gregor et al., 2016) | ≤ 3.58       |
| Real NVP (Dinh et al., 2016)  | 3.49          |
| Matnet + AR (Bachman, 2016)   | ≤ 3.24        |
| PixelCNN (van den Oord et al., 2016b) | 3.14          |
| VAE with IAF (Kingma et al., 2016) | ≤ 3.11    |
| Gated PixelCNN (van den Oord et al., 2016a) | 3.03          |
| PixelRNN (van den Oord et al., 2016b) | 3.00          |
| **Grayscale LatentPixelCNN** (this paper) | ≤ 2.98        |
| DenseNet VLAE (Chen et al., 2017) | ≤ 2.95        |
| PixelCNN++ (Salimans et al., 2017) | 2.92          |

Table 1. The negative log-likelihood of the different models for the CIFAR-10 test set measured as bits-per-dimension.
Figure 2. Random quantized grayscale samples from $p(\tilde{X})$ (top) and corresponding image samples from $p(X|\tilde{X})$ (bottom). The grayscale samples show several recognizable objects, which are subsequently also present in the color version.

Figure 3. CIFAR-10 images in original color (left) and quantized to 4-bit grayscale (center). Images sampled from our conditional model $p(X|\tilde{X})$, using the grayscale CIFAR images as latent variables (right). The images produced by our model are visually as plausible as the original ones.
4.2. Pyramid LatentPixelCNN.

**Experimental setup.** We evaluate the Pyramid LatentPixelCNN on the task of modeling face images. We rely on the \textit{aligned&cropped CelebA} dataset (Liu et al., 2015) that contains approximately 200,000 images of size 218x178. In order to focus human faces and not background, we preprocess all images in the dataset by applying a fixed 128x128 crop (left margin: 25 pixels, right margin: 25 pixel, top margin: 50, bottom margin: 40 pixels). We use a random 95\% subset of all images as training set and the remaining images as a test set.

For the Pyramid LatentPixelCNN we apply the latent variable decomposition 4 times. This results in a sequences of probabilistic models, where the first model generates faces in 8x8 resolution and the remaining ones gradually perform upsampling until reaching the final 128x128 resolution.

We use a PixelCNN++ architecture without down or upsampling layers with only 3 residual blocks to model the distributions \( p_0(\hat{X}) \) and \( p_0(X|\hat{X}) \) for all scales. For the embedding \( f_w(\hat{X}) \) we use a PixelCNN++ architecture with 15 residual blocks with downsampling layer after the residual block number 3 and upsampling layers after the residual blocks number 9 and 12. For all convolutional layers we set the number of filters to 100.

In the Adam optimizer we use an initial learning rate 0.001, a batch size of 16 and a learning rate decay of 0.999995.

**Modeling performance.** The Pyramid LatentPixelCNN achieves an upper bound on the negative log-likelihood of 1.52 bits-per-dimension. This much lower than for CIFAR-10 image, indicating that the face images have a simpler structure and higher resolution, which makes pixel values more correlated and easier to predict on average.

Before demonstrating and discussing face samples produced by our model we make an interesting observation. Recall that the output of the Pyramid LatentPixelCNN is a mixture of logistic distributions. We observe an intriguing effect related to the mixture representation of the predicted pixel distributions for face images: the perceptual quality of sampled faces substantially increases if we artificially reduce the predicted variance of the mixture components. We illustrate this effect in Figure 4, where we alter the variance by subtracting constants from a fixed set of \{0.0, 0.1, \ldots, 1.0\} from the predicted log-variance of the mixture components. Inspired by this observation we propose an alternative sampling procedure: for each pixel, we randomly sample one of the logistic components based on their weight in the predicted mixture. Then, we use the mode of this component as sampled pixel value, instead of performing a second random sampling step. This sampling procedure can be seen as a hybrid of probabilistic sampling and maximum a posteriori (MAP) prediction.

Figure 5 shows further samples obtained by MAP sampling. The produced images have very high perceptual quality, with some generated faces appearing almost photorealistic. The complete multi-scale sampling mechanism of the Pyramid LatentPixelCNN, from 8x8 to 128x128 images, is demonstrated in Figure 6.

**Discussion.** First, we observe that, despite the very high resolution of modeled images, the produced samples capture global human face characteristics, such as arrangement of face elements and global symmetries. At the same time, the set of samples is diverse, containing male as well as female faces, different hair and skin colors as well as facial expressions and head poses. Second, we emphasize that by properly decomposing the model we are able to scale the Pyramid LatentPixelCNN to produce samples with very high resolution of 128x128. As discussed previously in Section 3, this results from the fact that our decomposition allows to parametrize autoregressive parts of the image model by a light-weight architecture. Concretely, on an NVidia TitanX GPU, our Pyramid LatentPixelCNN without caching optimizations requires approximately 0.004 seconds on average to generate one image pixel, while a PixelCNN++ even with recently suggested caching optimizations requires roughly 0.05 seconds for the same task. Of course, if we add caching optimizations to our model its speed up should be even better.

5. Conclusion

In this paper we presented LatentPixelCNNs, an improved autoregressive probabilistic technique that includes latent variables into PixelCNN-type models. We derived two generative image models that exploit different image views as latent variables and address known limitations of existing PixelCNN models. The use of quantized grayscale images as latent variables resulted in a probabilistic model that captures global structure of very complex natural images and produces globally coherent samples. With multi-scale image views as latent variable, the model was able to efficiently produce nearly photo-realistic high-resolution images of human faces. Note, that these improvements are complementary and we plan to combine and extend them in a future work.

Furthermore, we gained interesting insights into the image modeling problem. First, our experiments suggest that texture and other low-level image information distract probabilistic models from focusing on more essential high-level image information, such as object shapes. Thus, it is beneficial to decouple the modeling of low and high-level image details. Second, we demonstrate that multi-scale image model, even with very shallow PixelCNN architectures, can accurately model high-resolution images.
Figure 4. Effect of the variance reduction. Numbers on top of each column indicates the amount of reduction in the predicted log-variance of the mixture components. The last column corresponds to MAP sampling.

Figure 5. Images sampled from the Pyramid LatentPixelCNN by MAP sampling. The generated faces are of very high quality, many being close to photorealistic. At the same time, the set of sample is diverse in terms of the depicted gender, skin color and head pose.

Figure 6. Visualization of the Pyramid LatentPixelCNN sampling process. Faces are first generated on a small, 8x8, resolution and then are gradually upsampled until reaching the desired 128x128 resolution.
References

Bachman, Philip. An architecture for deep, hierarchical generative models. In Conference on Neural Information Processing Systems (NIPS), 2016.

Carlsson, Gunnar, Ishkhanov, Tigran, De Silva, Vin, and Zomorodian, Afra. On the local behavior of spaces of natural images. International Journal of Computer Vision (IJCV), 76(1):1–12, 2008.

Chen, Xi, Kingma, Diederik P, Salimans, Tim, Duan, Yan, Dhariwal, Prafulla, Schulman, John, Sutskever, Ilya, and Abbeel, Pieter. Variational lossy autoencoder. International Conference on Learning Representations (ICLR), 2017.

Dahl, Ryan, Norouzi, Mohammad, and Shlens, Jonathon. Pixel recursive super resolution. arXiv preprint arXiv:1702.00783, 2017.

Denton, Emily L, Chintala, Soumith, Fergus, Rob, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In Conference on Neural Information Processing Systems (NIPS), 2015.

Dinh, Laurent, Krueger, David, and Bengio, Yoshua. NICE: Non-linear independent components estimation. arXiv preprint arXiv:1410.8516, 2014.

Dinh, Laurent, Sohl-Dickstein, Jascha, and Bengio, Samy. Density estimation using real NVP. arXiv preprint arXiv:1605.08803, 2016.

Geman, Stuart and Geman, Donald. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI), 6(6):721–741, 1984.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial networks. In Conference on Neural Information Processing Systems (NIPS), 2014.

Gregor, Karol, Danihelka, Ivo, Graves, Alex, Rezende, Danilo, and Wierstra, Daan. Draw: A recurrent neural network for image generation. In International Conference on Machine Learning (ICML), 2015.

Gregor, Karol, Besse, Frederic, Jimenez Rezende, Danilo, Danihelka, Ivo, and Wierstra, Daan. Towards conceptual compression. In Conference on Neural Information Processing Systems (NIPS), 2016.

Gulrajani, Ishaan, Kumar, Kundan, Ahmed, Faruk, Taiga, Adrien Ali, Visin, Francesco, Vazquez, David, and Courville, Aaron. PixelVAE: A latent variable model for natural images. International Conference on Learning Representations (ICLR), 2017.

Hyvärinen, Aapo, Hurri, Jarmo, and Hoyer, Patrick O. Natural Image Statistics: A Probabilistic Approach to Early Computational Vision., volume 39. Springer, 2009.

Kingma, Diederik P. and Ba, Jimmy. Adam: A method for stochastic optimization. In International Conference on Learning Representations (ICLR), 2014.

Kingma, Diederik P and Welling, Max. Auto-encoding variational bayes. In International Conference on Learning Representations (ICLR), 2014.

Kingma, Diederik P., Salimans, Tim, and Welling, Max. Improving variational inference with inverse autoregressive flow. Conference on Neural Information Processing Systems (NIPS), 2016.

Krizhevsky, Alex and Hinton, Geoffrey. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

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

Olshausen, Bruno A and Field, David J. Natural image statistics and efficient coding. Network: computation in neural systems, 7(2):333–339, 1996.

Ramachandran, Prajit, Paine, Tom Le, Khorrami, Pooya, Babaeizadeh, Mohammad, Chang, Shiyu, Zhang, Yang, Hasegawa-Johnson, Mark, Campbell, Roy, and Huang, Thomas. Fast generation for convolutional autoregressive models. Technical report, https://github.com/PrajitR/fast-pixel-cnn, 2017.

Reed, Scott E, Akata, Zeynep, Mohan, Santosh, Tenka, Samuel, Schiele, Bernt, and Lee, Honglak. Learning what and where to draw. In Conference on Neural Information Processing Systems (NIPS), 2016.

Roth, Stefan and Black, Michael J. Fields of experts: A framework for learning image priors. In Conference on Computer Vision and Pattern Recognition (CVPR), 2005.

Ruderman, Daniel L and Bialek, William. Statistics of natural images: Scaling in the woods. Physical review letters, 73(6):814, 1994.

Salimans, Tim, Goodfellow, Ian, Zaremba, Wojciech, Cheung, Vicki, Radford, Alec, and Chen, Xi. Improved techniques for training GANs. In Conference on Neural Information Processing Systems (NIPS), 2016.

Salimans, Tim, Karpathy, Andrej, Chen, Xi, Knigma, Diederik P., and Bulatov, Yaroslav. PixelCNN++: A PixelCNN implementation with discretized logistic mixture likelihood and other modifications. International Conference on Learning Representations (ICLR), 2017.
Sohl-Dickstein, Jascha, Weiss, Eric, Maheswaranathan, Niru, and Ganguli, Surya. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015.

van den Oord, Aaron and Schrauwen, Benjamin. Factoring variations in natural images with deep gaussian mixture models. In *Conference on Neural Information Processing Systems (NIPS)*, 2014.

van den Oord, Aaron, Kalchbrenner, Nal, Espeholt, Lasse, Vinyals, Oriol, and Graves, Alex. Conditional image generation with pixelCNN decoders. In *Conference on Neural Information Processing Systems (NIPS)*, 2016a.

van den Oord, Aaron, Kalchbrenner, Nal, and Kavukcuoglu, Koray. Pixel recurrent neural networks. *International Conference on Machine Learning (ICML)*, 2016b.

Wang, Xiaolong and Gupta, Abhinav. Generative image modeling using style and structure adversarial networks. In *European Conference on Computer Vision (ECCV)*, 2016.

Zhu, Song Chun, Wu, Yingnian, and Mumford, David. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision (IJCV)*, 27(2):107–126, 1998.

Zoran, Daniel and Weiss, Yair. From learning models of natural image patches to whole image restoration. In *International Conference on Computer Vision (ICCV)*, 2011.