# PIXELVAE: A LATENT VARIABLE MODEL FOR NATURAL IMAGES

**Ishaan Gulrajani**
University of Montreal

**Kundan Kumar**
University of Montreal
IIT Kanpur

**Faruk Ahmed**
University of Montreal

**Adrien Ali Taiga**
University of Montreal
CentraleSupelec

**Francesco Visin**
University of Montreal

**David Vazquez**
University of Montreal
Universitat Autonoma de Barcelona

**Aaron Courville**
University of Montreal
CIFAR Fellow

## ABSTRACT

Natural image modeling is a landmark challenge of unsupervised learning. Variational Autoencoders (VAEs) learn a useful latent representation and generate samples that preserve global structure but tend to suffer from image blurriness. PixelCNNs model sharp contours and details very well, but lack an explicit latent representation and have difficulty modeling large-scale structure in a computationally efficient way. In this paper, we present PixelVAE, a VAE model with an autoregressive decoder based on PixelCNN. The resulting architecture achieves state-of-the-art log-likelihood on binarized MNIST. We extend PixelVAE to a hierarchy of multiple latent variables at different scales; this hierarchical model achieves competitive likelihood on 64x64 ImageNet and generates high-quality samples on LSUN bedrooms.

## 1 INTRODUCTION

Building high-quality generative models of natural images has been a long standing challenge. Although recent work has made significant progress (Kingma & Welling, 2014; van den Oord et al., 2016a;b), we are still far from generating convincing, high-resolution natural images.

Many recent approaches to this problem are based on an efficient method for performing amortized, approximate inference in continuous stochastic latent variables: the variational autoencoder (VAE) (Kingma & Welling, 2014) jointly trains a top-down decoder generative neural network with a bottom-up encoder inference network. VAEs for images typically use rigid decoders that model the output pixels as conditionally independent given the latent variables. The resulting models learn a useful latent representation of the data and are usually effective at modeling global structure in images, but have difficulty capturing small-scale features such as textures and sharp edges due to the conditional independence of the output pixels (see Figure 2 middle), which significantly hurts both log-likelihood and quality of generated samples compared to other models.

PixelCNNs (van den Oord et al., 2016a;b) are another state-of-the-art image model. Unlike VAEs, PixelCNNs model image densities autoregressively, pixel-by-pixel. This allows the PixelCNN to capture fine details in images, as features such as edges can be precisely aligned. By leveraging carefully constructed masked convolutions (van den Oord et al., 2016b), PixelCNN can be trained efficiently in parallel on GPUs.

Nonetheless, PixelCNN models are still very computationally expensive. Unlike typical convolutional architectures they do not apply downsampling between layers, which means that each layer is computationally expensive and that the depth of a PixelCNN must grow linearly with the size of the images in order for it to capture dependencies between far-away pixels.

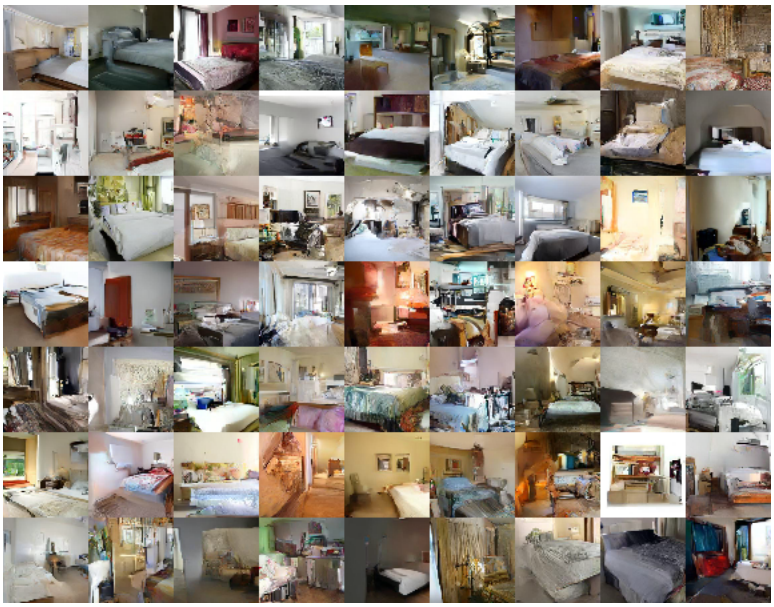

Figure 1: Samples from hierarchical PixelVAE on the LSUN bedrooms dataset.

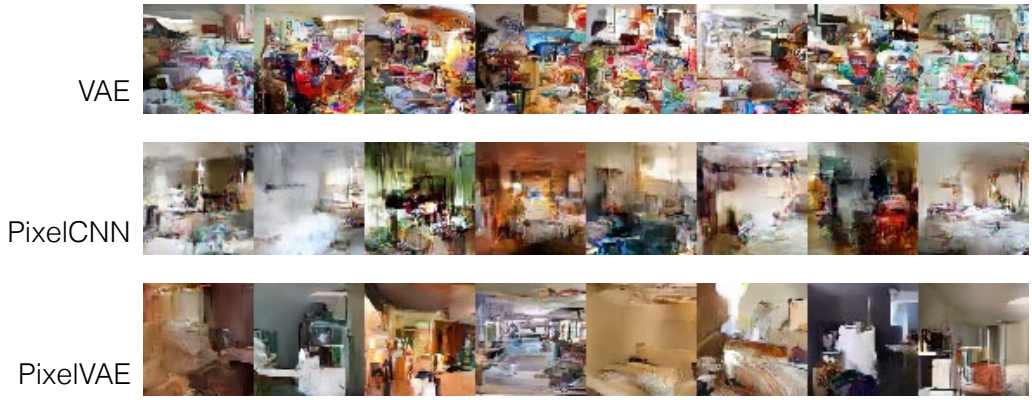


Figure 2: Samples from PixelCNN (top), convolutional VAE (middle), and PixelVAE (bottom) models of comparable size. VAE samples are generally blurry but convey a sense of objectness; the PixelCNN, on the other hand, generates sharp-looking textures but lacks clear global semantics. PixelVAE samples combine the sharpness and global coherence of both models.

PixelCNNs also do not explicitly learn a latent representation of the data, which can be useful for downstream tasks such as semi-supervised learning.

Our main contribution is a latent variable model that combines the advantages of VAEs and PixelCNNs. Specifically, we augment the decoder of a VAE with one or more PixelCNN layers, allowing it to model the output autoregressively rather than predicting the whole image at once. The use of an autoregressive decoder frees the latent variables from having to encode precise local alignment information, improving performance and enabling sharp generations (see Figure 1). We extend the model to a hierarchy of latent variables, allowing us to apply PixelCNN not just to the pixels of the output but also to the latent feature maps at each level. We evaluate our model on MNIST, 64x64 ImageNet, and the LSUN bedrooms dataset, achieving competitive log-likelihood scores on MNIST and generating samples with clear structure at multiple scales on LSUN bedrooms. Finally, we show that our model learns meaningful latent representations, which could make our model's representations a good foundation for semi-supervised learning.

## 2 RELATED WORK

There has been significant recent work on generative models for images, with two major lines of approach, namely variational inference based frameworks and adversarial models. We briefly discuss some of the most prominent members of both families below, especially those that are related to our approach.

The Variational Autoencoder (VAE) (Kingma & Welling, 2014) is an elegant framework to perform approximate variational inference by using neural networks to model both the approximate posterior (with an isotropic Gaussian prior) as well as the distribution of the data conditioned on the latent representation. Thanks to the reparameterization trick, this reduces to an end-to-end SGD-trainable autoencoder architecture that optimizes a lower bound estimate of the marginal likelihood of the data.

The concept of normalizing flows for stochastic gradient variational inference (Rezende & Mohamed, 2015) is applied to the VAE in Kingma et al. (2016) to allow a more flexible approximation of the posterior. The autoregressive formulation of the approximate posterior (following MADE (Germain et al., 2015)) allows for modeling nonlinear dependencies between elements of the latent space.

In the same family, the DRAW model (Gregor et al., 2015) uses instead a recurrent network encoder and a recurrent network decoder coupled with an attention mechanism. This makes the generation process sequential, thus allowing the model to improve the quality of the samples over time in an iterative fashion.

The key member of the other line of approach, i.e., the adversarial models, is the Generative Adversarial Network (GANs) (Goodfellow et al., 2014) which pits a generator network and a discriminator network against each other. The generator tries to generate samples similar to the training data to fool the discriminator, and the discriminator tries to detect if the sample originates from the data distribution or not. GANs are known for providing samples that are qualitatively the best, yet they have some downsides: it is non-trivial to derive a data likelihood (Parzen-window based estimates are usually used to this end) and they exhibit unstable training dynamics. Recent works along this direction have improved the stability in training (Salimans et al., 2016), and scaled up the size of the samples – as well as improved their quality – through the application of CNNs (with upsampling) (Radford et al., 2015).

The idea of using latent representations to capture global dependence while modeling the output space in a decomposed fashion has been explored in the context of sentence modeling in Bowman et al. (2016), that demonstrates the effectiveness of a stochastic latent layer to capture global semantics while modeling local structure with RNNs for generating sentences.

## 3 PIXELVAE MODEL

Like a VAE, our model jointly trains an "encoder" inference network, which maps an image $x$ to a posterior distribution over latent variables $z$, and a "decoder" generative network, which models a distribution over $x$ conditioned on $z$. The encoder and decoder networks are composed of a series of convolutional layers, respectively with strided convolutions for downsampling in the encoder and transposed convolutions for upsampling in the decoder.

As opposed to most VAE decoders model each dimension of the output independently (for example, by modeling the output as a Gaussian with diagonal covariance), we use a conditional PixelCNN in the decoder. Our decoder models $x$ as the product of each dimension $x_i$ conditioned on all previous dimensions and the latent variable $z$:

$$p(x|z) = \prod_i p(x_i|x_1, \ldots, x_{i1}, z) \tag{1}$$

We first transform $z$ through a series of convolutional layers into featuremaps with the same spatial resolution as the output image and then concatenate the resulting featuremaps with the image.

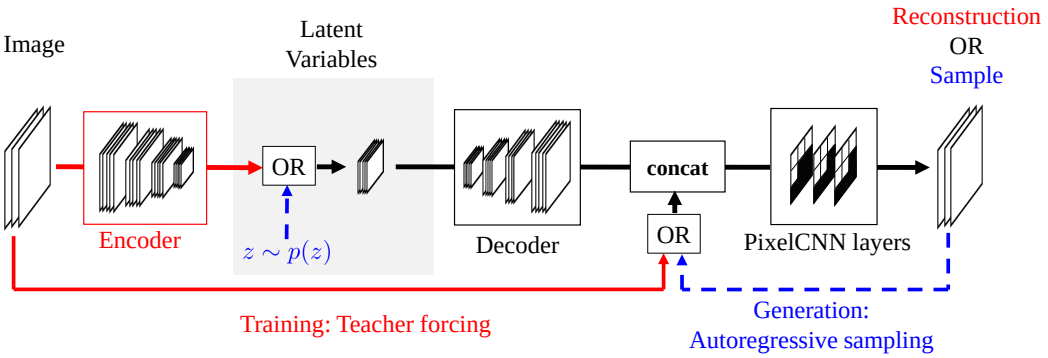


Figure 3: Our proposed model, PixelVAE, makes use of PixelCNN to model an autoregressive decoder for a VAE. VAEs, which assume independence among pixels, are known to suffer from blurry samples, while PixelCNN, modeling the joint distribution, produces sharp samples, but lack a latent representation that might be more useful for downstream tasks. PixelVAE combines the best of both worlds, providing a meaningful latent representation, while producing sharp samples.

The resulting concatenated featuremaps are then further processed by several PixelCNN masked convolutional layers and a final PixelCNN 256-way softmax output.

Unlike typical PixelCNN implementations, we use very few PixelCNN layers in our decoder, relying on the latent variables to model the structure of the input at scales larger than the combined receptive field of our PixelCNN layers. As a result of this, our architecture captures global structure at a much lower computational cost than a standard PixelCNN implementation.

### 3.1 Multi-scale architecture

The performance of VAEs can be improved by stacking them to form a hierarchy of stochastic hidden layers: in the simplest configuration, the VAE at each level models a distribution over the latent variables at the next level downward, with generation proceeding downward and inference upward through each level. In convolutional architectures, the intermediate latent variables are typically organized into featuremaps whose spatial resolution decreases toward higher levels.

Our model can be extended in the same way. When we do this, at each level, the generator is a conditional PixelCNN over the latent features in the next level downward. This lets us autoregressively model with PixelCNN not only the output distribution over pixels but also the prior over each set of latent featuremaps. The higher-level PixelCNN decoders use diagonal Gaussian output layers instead of 256-way softmax, and model the dimensions within each spatial location independently (this is done for simplicity, but is not a limitation of our model).

For a model with L levels of latent variables, we train this model by minimizing the negative of the evidence lower bound:

$$\begin{aligned} -L(x, \theta_0, ..., \theta_L, \phi_0, ..., \phi_L) = &- \log p_{\theta_0}(x|z_1) + D_{KL}(q_{\phi_0}(z_1|x)||p_{\theta_1}(z_1|z_2)) \\ &+ \sum_{l=2}^{L} D_{KL}(q_{\phi_1}(z_l|z_{l-1})||p(z_l)) \end{aligned} \tag{2}$$

where $\theta$ are the decoder parameters and $\phi$ are the encoder parameters.

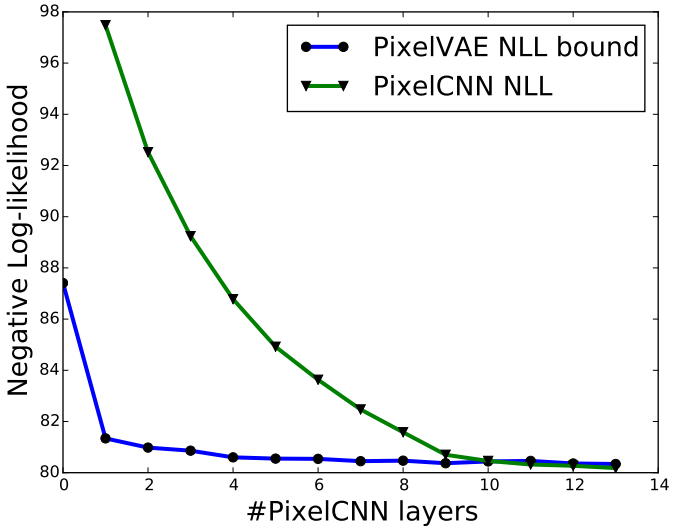


Figure 4: Comparison of NLL upper bound of PixelVAE and NLL for PixelCNN as a function of the number of PixelCNN layers used.

# 4 EXPERIMENTS

## 4.1 MNIST

We evaluate our model on the binarized MNIST dataset (Salakhutdinov & Murray, 2008; Lecun et al., 1998) and report results in Table 1. We also experiment with a variant of our model in which each PixelCNN layer is directly conditioned on a linear transformation of latent variable, $z$ (rather than transforming $z$ first through several upsampling convolutional layers) (as in van den Oord et al. (2016b) and find that this further improves performance, achieving an NLL upper bound comparable with the current state of the art. We estimate the marginal NLL of our model (using 50 importance samples per datapoint) and find it achieves state of the art performance.

One advantage of our architecture is that we can achieve strong performance with very few PixelCNN layers, which make training and sampling expensive. To demonstrate this, we compare the performance of our model to PixelCNN as a function of the number of PixelCNN layers (figure **??**). We can see that adding a single PixelCNN layer has a dramatic impact on NLL bound of PixelVAE. This is what we expect since the additional PixelCNN layer helps model the local characteristics very well which is complementary to the global characteristics which VAE with no auto-regressive layer models. We also find that with fewer number of PixelCNN layers, our conditional PixelVAE decoder does much better than the unconditional PixelCNN model. In our experiments, we have used PixelCNN layers with no blind spots using vertical and horizontal stacks as proposed in van den Oord et al. (2016b).

In figure 5, we compare the NLL bound breakdown between the reconstruction cost and KL cost for different number of PixelCNN layers. There is sharp drop in Kl-divergence part of cost when we use a single auto-regressive layer compared to no auto-regressive layer. The more information latent codes store about the data, the further their distribution will be from the prior imposed on them. KL-divergence term represents this deviation of the latent code distribution from its prior distribution. Hence, the sharp drop in KL-divergence cost signifies that the burden on latent codes to store all information required for reconstruction of the sample has been significantly reduced. Furthermore, since addition of a single PixelCNN layer allows to model interactions between pixels which are atmost 2 pixels away from it (since our masked convolution filter size is 5x5), we can say that the latent code has been freed from storing local pixel interactions and hence will model more global structures.

| Models | NLL test |
|---|---|
| PixelCNN van den Oord et al. (2016a) | 81.3 |
| VAE | $\leq$87.4 |
| PixelVAE | $\leq$80.64 |
| GatedPixelCNN | 80.1 |
| GatedPixelVAE | $\leq$80.08 |
| PixelRNN van den Oord et al. (2016a) | 79.20 |
| GatedPixelVAE without upsampling | $\approx$**79.17** ($\leq$ 79.77) |

Table 1: Comparison of performance of different models on binarized MNIST. PixelCNN is the model described in van den Oord et al. (2016a) with masked convolutions and relu activations. Our corresponding latent variable model is PixelVAE. GatedPixelVAE uses the GatedPixelCNN activation function in van den Oord et al. (2016b). GatedPixelVAE without upsampling is the model where a linear transformation of latent variable conditions the (gated) activation in every PixelCNN layer instead of using upsampling layers.

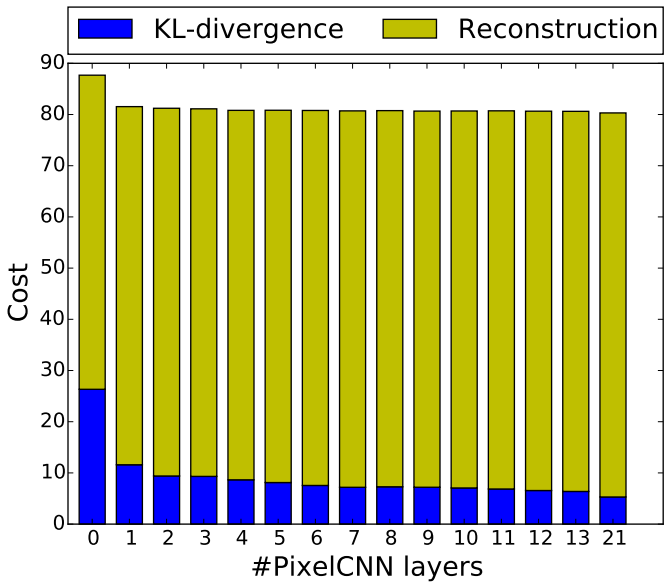


Figure 5: NLL break down into KL divergence and reconstruction cost.

### 4.2 LSUN BEDROOMS

To evaluate our model's performance with more data and complicated image distributions, we perform experiments on the LSUN bedrooms dataset (Yu et al., 2015). We use the same preprocessing as in Radford et al. (2015) to remove duplicate images in the dataset. For quantitative experiments we use a 32x32 downsampled version of the dataset, and we present samples from a model trained on the 64x64 version.

We train a two-level PixelVAE with latent variables at 1x1 and 8x8 spatial resolutions. We find that this outperforms both a two-level convolutional VAE with diagonal Gaussian output and a single-level PixelVAE in terms of log-likelihood and sample quality. We also try replacing the PixelCNN layers in the higher level with a diagonal Gaussian decoder and find that this hurts log-likelihood, which suggests that multi-scale PixelVAE uses those layers effectively to autoregressively model latent features.

To see which features are modeled at each scale, we draw multiple samples while varying the sampling noise at only a specific level of latent variables (Figure 6). Samples with all latent variables in common are almost indistinguishable and differ only in precise positioning and shading details, indicating that the model uses the pixel-level autoregressive distribution to model only these features. Samples with only varied middle-level latent variables have different objects and colors, but appear

to have similar basic room geometry. Finally, samples with varied top-level latent variables have diverse room geometry.

### 4.3 64X64 IMAGENET

We evaluate our model on the 64x64 ImageNet dataset used in (van den Oord et al., 2016a), and find our model achieves performance competitive with the state of the art. We report validation set likelihood in table 2.

| Model | NLL val (train) |
|---|---|
| PixelRNN van den Oord et al. (2016a) | 3.63 (3.57) |
| Gated PixelCNN van den Oord et al. (2016b) | 3.57 (3.48) |
| Hierarchical PixelVAE | $\leq$3.66 (3.59) |

Table 2: Model performance on 64x64 ImageNet.

Figure 6: We visually inspect the variation in image features captured by the different levels of stochasticity in our model. For the two-level latent variable model trained on $64 \times 64$ LSUN bedrooms, *(top)* we hold the mid-level and pixel-level sampling noise constant, *(middle)* we hold the top-level and pixel-level sampling noise constant, varying only the middle level, *(bottom)* latent variables are held constant, varying the noise only at the pixel-level. It appears that the top-level latent variables learn to model room structure and overall geometry, the middle-level latents model color and texture features, and the pixel-level distribution models low-level image characteristics such as texture, alignment, shading.

## 5 CONCLUSIONS

In this paper, we introduced a latent variable model for natural images with autoregressive decoder. We empirically validated the image modelling capability of the model on challenging LSUN and

Imagenet datasets. Relative to the standard VAE, the advantage brought by the introduction of the autoregressive decoder is a significant improvement in generated samples quality. We establish a new state-of-the-art on binarized MNIST dataset in terms of likelihood and demonstrate that our model generates high-quality samples on LSUN bedrooms.

## ACKNOWLEDGMENTS

The authors would like to thank the developers of Theano Theano Development Team (2016). We acknowledge the support of the following agencies for research funding and computing support: Ubisoft, Nuance Foundation, NSERC, Calcul Quebec, Compute Canada, CIFAR, MEC Project TRA2014-57088-C2-1-R, SGR project 2014-SGR-1506 and TECNIOspring-FP7-ACCI grant.

## REFERENCES

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. 2016.

Matthieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. *CoRR*, abs/1502.03509, 2015. URL `https://arxiv.org/abs/1502.03509`.

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

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

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

Diederik P. Kingma, Tim Salimans, and Max Welling. Improving variational inference with inverse autoregressive flow. *CoRR*, abs/1606.04934, 2016.

Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*, 2015.

Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *In Proceedings of the 25th international conference on Machine learning*, 2008.

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. URL `http://arxiv.org/abs/1605.02688`.

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

Aäron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders. *CoRR*, abs/1606.05328, 2016b. URL `http://arxiv.org/abs/1606.05328`.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.