PixelPyramids: Exact Inference Models from Lossless Image Pyramids

Shweta Mahajan[1]   Stefan Roth[1,2]

[1]Department of Computer Science, TU Darmstadt   [2] hessian.AI

Abstract

Autoregressive models are a class of exact inference approaches with highly flexible functional forms, yielding state-of-the-art density estimates for natural images. Yet, the sequential ordering on the dimensions makes these models computationally expensive and limits their applicability to low-resolution imagery. In this work, we propose Pixel-Pyramids,[1] *a block-autoregressive approach employing a lossless pyramid decomposition with scale-specific representations to encode the joint distribution of image pixels. Crucially, it affords a sparser dependency structure compared to fully autoregressive approaches. Our PixelPyramids yield state-of-the-art results for density estimation on various image datasets, especially for high-resolution data. For CelebA-HQ* 1024×1024*, we observe that the density estimates (in terms of bits/dim) are improved to* $\sim 44\%$ *of the baseline despite sampling speeds superior even to easily parallelizable flow-based models.*

1. Introduction

Deep generative models have enabled significant progress in capturing the probability density of highly structured, complex data, such as of natural images [5, 16, 27, 34, 45] or raw audio [11, 21, 47]. These models find application in a wide range of tasks including image compression [8, 44], denoising [2], inpainting [42, 46], and super-resolution [6, 30]. Popular generative models for images include Generative Adversarial Networks (GANs) [16], which transform a noise distribution to the desired distribution, and Variational Autoencoders (VAEs) [27], where the data is modeled in a low-dimensional latent space by maximizing a lower bound on the log-likelihood of the data [36]. However, GANs are not designed to provide exact density estimates and VAEs only approximate the underlying true distribution with intractable likelihoods, posing challenges in both training and inference.

In contrast, autoregressive models [14, 32, 45, 46] and normalizing flows [3, 12, 13, 18, 25] are exact inference approaches, which estimate the exact likelihood of the data.

---

[1]Code available at https://github.com/visinf/pixelpyramids
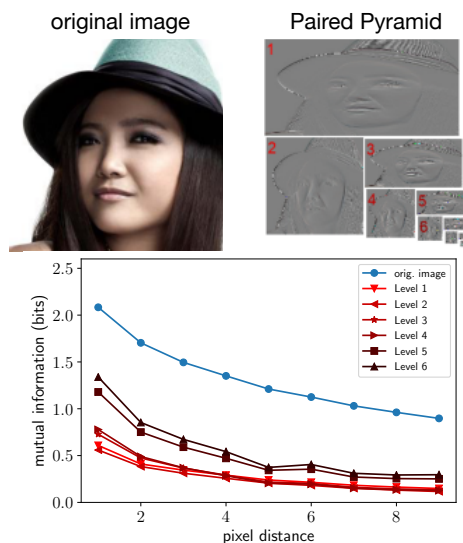

Figure 1. *Motivation for PixelPyramids.* An image from CelebA [23] *(top left)* is decomposed into a Paired Pyramid [43] *(top right)* through an invertible, lossless mapping. Considering the mutual information between pixels as a function of pixel distance on the CelebA dataset *(bottom)*, there is significant spatial dependence in the original image *(blue, circles)*, even between distant pixels. The fine components of the Paired Pyramid at different levels *(other lines)* have significantly more localized dependencies, decreasing further as the resolution increases. Our PixelPyramids exploit this for efficient block-autoregressive generative image modeling.

Autoregressive models factorize the joint target distribution into a product of conditional distributions with a certain ordering over the dimensions, thereby encoding complex dependencies in the data distribution for effective density estimation. However, the sequential dependency structure makes efficient parallelization difficult. Normalizing flows, on the other hand, map the input data to a known base distribution through a series of invertible transformations, providing efficient sampling. However, the invertibility constraint limits their expressiveness and their density estimation performance lags behind that of autoregressive models.

Recent work [31, 35, 50] aims to improve the computationally expensive (autoregressive) or constrained (normalizing flows) exact inference models through multi-scale feature representations of images, such as that from Wavelet

or pyramid representations. The key idea is that instead of directly modeling the complex image distribution, it is decomposed into a series of simpler pixel representations, which can be encoded with comparatively less complex functional forms of exact inference models. However, these approaches either rely on a specific design choice of pixel orderings to reduce autoregressive connections or contain higher quantization levels, which makes encoding difficult.

In this work, we propose *PixelPyramids* – an expressive and computationally efficient block-autoregressive model for the discrete joint distribution of pixels in images. We make the following contributions: *(i)* Our PixelPyramids leverage ideas from lossless image coding, specifically a low-entropy multi-scale representation based on so-called Paired Pyramids [43] (*cf*. Fig. 1, *top*), to encode images in a coarse-to-fine manner, where the conditional generative model at each pyramid scale is conditioned on the coarse(r) image from the previous level. The subsampled images and the scale-specific components are encoded in the same number of pixels and at the same quantization level as the original image, resulting in an efficient exact inference approach for density estimation.[2] *(ii)* The conditional generative model for the fine component at each scale exploits a U-Net architecture [37] to encode global context from the coarser scales. A Convolutional LSTM [40] is used to capture the spatial pixel dependencies within each fine component. By conditioning on the coarser scales, the spatial dependency structure of the fine component at each scale is more localized (*cf*. Fig. 1, *bottom*), especially as we go to higher resolutions in the pyramid. Thus, long-range dependencies at each scale can be modeled with fewer autoregressive steps in a computationally efficient setup.

Furthermore, *(iii)* a key advantage of PixelPyramids is that the fine components at different scales are conditionally independent of each other and can, therefore, be trained in parallel. This makes the model applicable for density estimation and synthesis of high-resolution images. PixelPyramids require $\mathcal{O}(\log N)$ sampling steps for an $N \times N$ image, which is significantly more efficient than the $\mathcal{O}(N^2)$ sequential steps of fully autoregressive approaches. Finally, *(iv)* we show that our PixelPyramids yield state-of-the-art density estimates and high-quality image synthesis on standard image datasets including CelebA-HQ [23], LSUN [49], and ImageNet [38], as well as on the high-resolution 1024×1024 CelebA-HQ dataset at a much lower computational cost than previous exact inference models.

2. Related Work

Exact inference models explicitly maximize the data log-likelihood, making them attractive for estimating multi-modal densities. Recent work on deep generative models

applicable to both density estimation and generation has focused on autoregressive methods and normalizing flows.

Normalizing flows map a simple base distribution to the complex data distribution through a series of invertible transformations with tractable Jacobian determinants [1, 12, 26]. Kingma *et al*. [25] extend the flow-based RealNVP model [13] with invertible 1×1 convolutions with applications to high-resolution imagery. Various work [18, 20, 31] improves flow-based models with architectural modifications to the non-linearities in the invertible transformations to increase their modeling capacity or provide for efficient training. While easy to parallelize and thus efficient, the representational power of flow-based models reflected in density estimates lags behind that of autoregressive models.

Autoregressive models factorize the image distribution into a product of conditionals over the pixels, exploiting the chain rule for exact inference [14, 17, 19, 33]. Deep neural generative autoregressive models, *e.g*. the PixelCNN and PixelRNN of van den Oord *et al*. [45, 46], parameterize the conditionals with convolutional or recurrent neural networks, achieving state-of-the-art density estimates on natural images. Kalchbrenner *et al*. [22] apply autoregressive models to raw audio. Subsequent work of Chen *et al*. and Salimans *et al*. [9, 39] introduces architectural enhancements to PixelCNN such as self-attention layers [9] and a discretized logistic mixture model [39]. One of the main limitations of these approaches is that generation proceeds through ancestral sampling, making them slow and difficult to parallelize. Song *et al*. [41] and Wiggers *et al*. [48] marginally improve the sampling speed by parallelizing the Jacobi updates with Gauss-Seidel and fixed-point iterations, respectively, subject to the availability of specific hardware resources. Noting the current limitations in modeling sequential dependencies in a computationally efficient manner, we introduce a multi-scale coarse-to-fine structure with localized dependency structures to model the discrete joint distribution of image pixels.

Multi-scale representations have been leveraged in GANs and VAEs for encoding the multimodality of the data distribution. Denton *et al*. [10] use residuals from Laplacian pyramids [7] for multi-stage generation in GANs. Dorta *et al*. [15] utilize detail coefficients from Laplacian pyramids in the latent space of VAEs for image manipulation. Razavi *et al*. [34] extend VQ-VAEs with hierachical low-level and high-level details in the latent space. In this work, we consider models that provide for exact inference in contrast.

Multiscale-PixelCNNs [35] aim to improve the sampling speed of autoregressive models with strong independence assumptions on the conditionals in the subsampled images. Menick *et al*. [32] synthesize by sequential conditioning on sub-images and employ an autoregressive structure not only along the spatial dimensions but also along image pixel values. Both methods rely on a design choice of the pixel or-
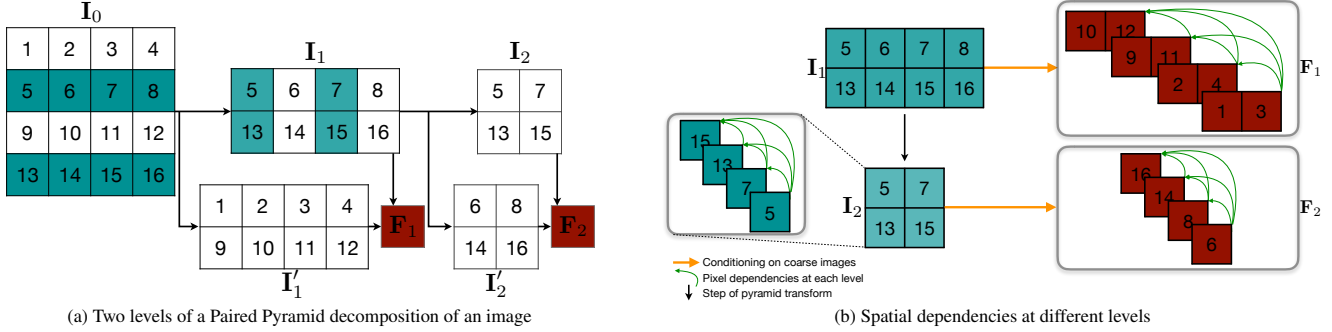
---

[2]We use the common term 'density estimation' even if we, strictly speaking, estimate a discrete distribution rather than a continuous density.

(a) Two levels of a Paired Pyramid decomposition of an image

(b) Spatial dependencies at different levels

Figure 2. *Paired Pyramids and the imposed dependency structure. (left)* Subsampling procedure in a Paired Pyramid (numbers denote pixel indices). In the first iteration, $\mathbf{I}_0$ is subsampled along the rows of the image, yielding $\mathbf{I}_1$ and $\mathbf{I}'_1$. The fine component $\mathbf{F}_1$ is the modulo difference of $\mathbf{I}'_1$ and $\mathbf{I}_1$, see Eq. (2). In the second iteration, $\mathbf{I}_1$ is subsampled along the columns, yielding $\mathbf{I}_2$ and $\mathbf{I}'_2$. *(right)* The spatial dependencies are modeled densely at the coarsest level (fully autoregressive) and more localized in pixel groups at higher, finer levels.

dering for generating multiple levels of the pyramid, which does not take into account image statistics such as harnessing scale-specific information. Kolesnikov *et al.* [28] introduce conditionals with grayscale image representations to capture global context. The conditionals in these multi-scale autoregressive models are applied directly in the pixel space and, therefore, have redundancies in the dependency structure at different scales. Recent work of Bhattacharyya *et al.* [4] and Yu *et al.* [50] increases the modeling flexibility of flow-based models with multi-scale coarse-to-fine scale-space representations based on Haar Wavelets. In our work, to improve the modeling flexibility and computational efficiency of autoregressive models, we leverage multi-scale lossless decompositions, which provide localized scale-specific representations and can be encoded with a limited number of autoregressive connections.

3. Block-Autoregressive PixelPyramids

To learn the complex joint distribution of image data with exact inference and generation in a computationally efficient framework, we formulate a low-entropy multi-scale block-autoregressive approach termed *PixelPyramids*. Our method leverages Paired Pyramids [43] from the literature on lossless image coding to decompose the image into low-entropy components with a local dependency structure, which makes it easier to perform density estimation.

3.1. Paired Pyramids

Proposed by Torbey and Meadows [43], Paired Pyramids are an invertible, lossless transformation of the image $\mathbf{I}_0$ of size $N_0 \times N_0 \times C$ into L levels of fine and coarse components. One of the spatial dimensions is reduced by a factor of two at each level of the image pyramid by alternatingly subsampling along the rows and columns of the image. To construct the *coarse component* $\mathbf{I}_i$ at level $i \in \{1, \ldots, L\}$, one row (column) from each of the non-overlapping pairs of adjacent rows (columns) of the previous level $\mathbf{I}_{i-1}$ is re-

tained without any transformation to form a subsampled image (see Fig. 2a). This reduces the spatial dimension by a factor of two, *i.e.* to size $N_{i-1}/2 \times N_{i-1} \times C$ when subsampled along the rows (analogously for subsampling along the columns). The image $\mathbf{I}'_i$ consisting of the remaining rows (columns) from each pair, also with size $N_{i-1}/2 \times N_{i-1} \times C$, is used to encode the *fine component* $\mathbf{F}_i$ through the *modulo difference* between the adjacent pairs of rows (columns) as

$$\mathbf{F}_i = (\mathbf{I}'_i - \mathbf{I}_i) \bmod 2^b. \tag{1}$$

Here, b is the number of bits for encoding the image $\mathbf{I}_0$, *i.e.* $\mathbf{I}_0 \in \{0, \ldots, 2^b - 1\}^{N_0 \times N_0 \times C}$, and its components $\mathbf{I}_i, \mathbf{I}'_i$, or $\mathbf{F}_i$. The modulo K difference, with $x, y \in \{0, \ldots, K-1\}$, can be computed as

$$(x - y) \bmod K = \begin{cases} (x - y) & \text{if } x \geq y \\ K - (y - x) & \text{otherwise.} \end{cases} \tag{2}$$

Thus, the modulo difference of the two subsampled images lies in the same range as the subsampled or the original image. Furthermore, the subsampled component $\mathbf{I}'_i$ can be fully reconstructed given the fine component $\mathbf{F}_i$ and the coarse component $\mathbf{I}_i$ using the inverse transformation

$$\mathbf{I}'_i = (\mathbf{F}_i + \mathbf{I}_i) \bmod 2^b. \tag{3}$$

Representational properties. Paired Pyramids have various favorable properties: *(i)* Using the modulo operation, the fine components can be represented at the *same quantization level* as the original image (no additional bit needed to represent the sign [43]). *(ii)* Moreover, the resulting pyramid representation has the *same number of pixels* as the original image (see Fig. 1, *top*). *(iii)* This makes the modulo difference-based pyramid transform an *invertible, lossless image coding scheme* [43]. Note that this is unlike the standard form of a Laplacian pyramid [7], which is overcomplete and the detail coefficients are difficult to represent in the same range and quantization level as the original
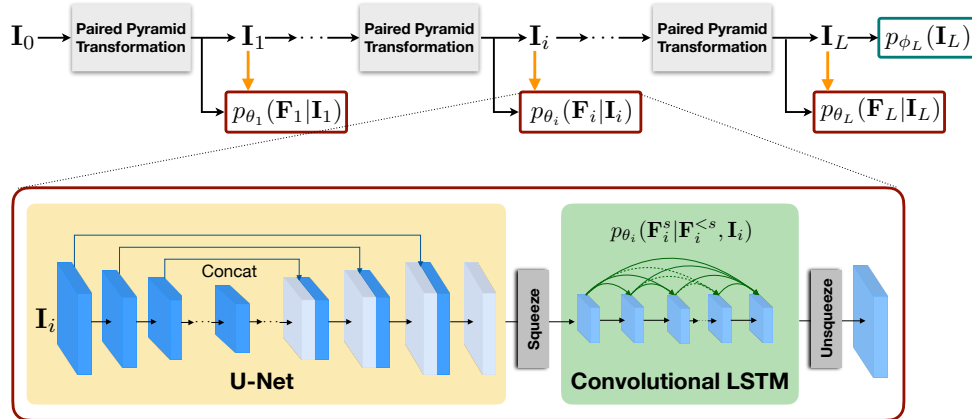
Figure 3. *PixelPyramids – Multi-scale block-autoregressive generative model based on Paired Pyramids* [43]. The proposed PixelPyramids rely on a U-Net [37] for capturing coarse-scale context, conditioned on which Conv-LSTMs [40] emit the fine components of the pyramid.
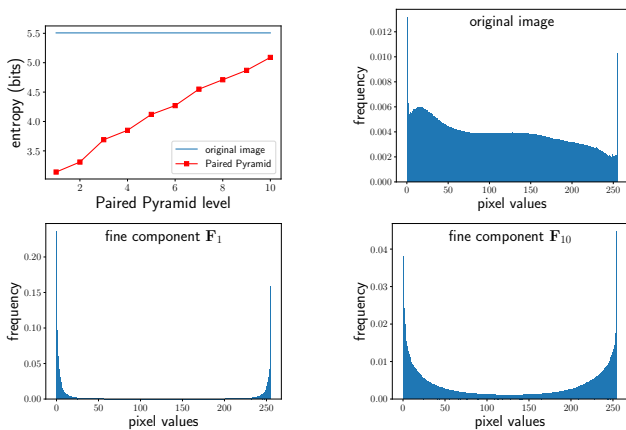

Figure 4. *Low-entropy decomposition.* Entropy values at different pyramid levels *(top left)*. Empirical marginal distributions on CelebA-HQ for the original pixel values *(top right)*, the pixel values of the fine component $\mathbf{F}_1$ *(bottom left)* and and the fine component $\mathbf{F}_{10}$ *(bottom right)*. Note the significantly reduced entropy from 5.51 bits for the original pixel values to 3.14 bits for $\mathbf{F}_1$.

image without loss of information. Here, we leverage the local, scale-specific representations of Paired Pyramids to develop a computationally efficient deep generative model with tractable likelihoods for complex image distributions.

Statistical properties. To see why Paired Pyramids are well suited also for generative modeling, we consider their statistical properties. We observe the following: *(i)* The fine components $\mathbf{F}_i$ at each level represent the scale-specific image structure and, as shown in Fig. 1 *(bottom)*, have *significantly less pixel-to-pixel correlations* than the original image. *(ii)* Furthermore, as shown in Fig. 4, its pixel values are highly concentrated around 0 and $2^b - 1$, *i.e. have low entropy* compared to the much more uniform marginal pixel distribution of the original images. Both the low entropy and the low spatial correlation make the fine components

substantially easier to model than the original image. *(iii)* Importantly, entropy and spatial correlation decrease further at lower (high res.) pyramid levels, which means that *the computationally most costly levels are the easiest to model.*

3.2. Pixel Pyramid Network

To define PixelPyramids (Fig. 3), we first construct a Paired Pyramid representation of the original image $\mathbf{I}_0$, consisting of L levels with coarse components $\{\mathbf{I}_1, \ldots, \mathbf{I}_L\}$ and fine components $\{\mathbf{F}_1, \ldots, \mathbf{F}_L\}$. Equipped with this lossless (one-to-one) decomposition, the probability distribution $p_{\theta, \phi_L}(\mathbf{I}_0)$ of images, parameterized by $\theta = \{\theta_1, \ldots, \theta_L\}$ and ϕ_L, autoregressively factorizes *w.l.o.g.* as

$$p_{\theta, \phi_L}(\mathbf{I}_0) = p_{\phi_L}(\mathbf{I}_L) \prod_{i=1}^{L} p_{\theta_i}(\mathbf{F}_i \mid \mathbf{I}_i). \qquad (4)$$

Thus, the image distribution can be obtained as the product of the distribution of the coarsest component $\mathbf{I}_L$ and the conditional distributions of the fine components $\mathbf{F}_i$ conditioned on the coarse component $\mathbf{I}_i$ at each level. The main advantage of this decomposition is three-fold: *(i)* The fine components have *low marginal entropy*, thus are easier to model using parameterized conditional deep generative models; *(ii)* the fine components also have *less spatial correlations* and represent scale-specific local information; *(iii)* the spatial correlations of the fine components are *less at the* (computationally challenging) *higher resolution levels* than at the (computationally lighter) lower resolution levels. Therefore, the conditional distribution $p_{\theta_i}(\mathbf{F}_i \mid \mathbf{I}_i)$ becomes easier to model with increasing scales i using computationally efficient models with local pixel dependencies. The factorization in Eq. (4) has yet another advantage: The conditional generative model of the fine component at every scale is independent of those at the other scales. Thus they can be trained in parallel. Taken together, this overcomes

the limitations of models like PixelCNN [45] or SPN [32], which rely on a fully autoregressive structure to capture the underlying multimodality and complex pixel dependencies.

Per-level conditionally generative models. In addition to conditioning on the coarse image $\mathbf{I}_i$, we introduce a partial autoregressive structure to better encode the dependencies within the fine component $\mathbf{F}_i$. To that end, $\mathbf{F}_i$ is further decomposed into subsampled images (see Sec. 3.3 for details), where we enforce sequential dependencies across the pixels of different subsampled images while the pixels within each subsampled image are conditionally independent of each other. This is illustrated in Fig. 2b *(top)* for the case of four such subsampled images. Let S_i be the number of subsampled images in the fine component at a certain level i and $\mathbf{F}_i^s, s \in \{1, \dots, S_i\}$ denote a subsampled image. Then the conditional distribution $p_{\theta_i}(\mathbf{F}_i \mid \mathbf{I}_i)$ decomposes into a product of conditionals using the product rule as

$$p_{\theta_i}(\mathbf{F}_i \mid \mathbf{I}_i) = p_{\theta_i}(\mathbf{F}_i^1 \mid \mathbf{I}_i) \prod_{s=2}^{S_i} p_{\theta_i}(\mathbf{F}_i^s \mid \mathbf{F}_i^{<s}, \mathbf{I}_i). \quad (5)$$

The benefit of the Paired Pyramid representation is that we do not need many such autoregressive sampling sub-images, since the fine components are spatially decorrelated, especially those at higher resolutions (see above).

Modeling the coarsest level. The image at the coarsest level L has size $N_0/2^{\lfloor (L+1)/2 \rfloor} \times N_0/2^{\lfloor L/2 \rfloor} \times C$ and represents the global structure at a much lower resolution than the original image. Assuming that we have sufficiently many levels, the coarsest level is small enough to adopt a fully autoregressive model where, similar to PixelCNN, pixels $\mathbf{I}_L^s$ are generated one-by-one using the chain rule, conditioning on both pixels in the rows above it and to the left within the same row, $\mathbf{I}_L^{<s}$ (Fig. 2b, *bottom*). We thus have

$$p_{\theta_L}(\mathbf{I}_L) = p_{\theta_L}(\mathbf{I}_L^1) \prod_{s=2}^{S_L} p_{\theta_L}(\mathbf{I}_L^s \mid \mathbf{I}_L^{<s}). \quad (6)$$

Per-pixel conditional distributions. The pixel values for the fine components at each level as well as the image at the coarsest level are modeled as discrete distributions. Note that due to the properties of the Paired Pyramid, all components have the same range and quantization as the original image. Similar to PixelCNN++ [39], we use a continuous mixture of M logistic distributions to estimate the conditional distribution of a pixel value $x \in \{0, \dots, 2^b - 1\}$ of the fine or coarse component as

$$p(x \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\omega}) = \sum_{i=1}^{M} \pi_i \Big[\sigma\big((x + 0.5 - \mu_i)/\omega_i\big) \\ - \sigma\big((x - 0.5 - \mu_i)/\omega_i\big) \Big]. \quad (7)$$

Here, $\sigma(\cdot)$ is the logistic function, μ_i and ω_i are the conditional mean and standard deviation of each mixture component with weight π_i. The mean of the green channel is taken to linearly depend on the value of the red sub-pixel and, analogously, the mean of the blue channel on the red and green sub-pixel values. For instantiating a pixel value from the continuous logistic mixture, the samples are rounded to the nearest discrete value with the same quantization as the fine component. We refer to [39] for more details.

3.3. Network architecture

To yield the conditional distributions $p_{\theta_i}(\mathbf{F}_i \mid \mathbf{I}_i)$, we first leverage a U-Net architecture [37] for capturing the conditional information from $\mathbf{I}_i$ (Fig. 3). Specifically, its low-resolution feature maps in the encoder provide global context from the conditioning pixels for encoding scale-specific image details. Combining this through skip connections with the upsampled features of the decoder adds local context at higher resolutions (see Sec. 4.1 for details). The feature map obtained from the final U-Net layer, with the same spatial dimensions as that of the fine component, is then used as input for modeling $\mathbf{F}_i$ following Eq. (5).

To that end, we decompose the image/feature map into subsampled images. Specifically, we apply a number of squeeze operations [12] to trade off spatial dimensions for channel dimensions. The squeeze operation stacks non-overlapping $2 \times 2 \times 1$ local neighborhoods along the feature channels by reshaping them into blocks of size $1 \times 1 \times 4$. Applying this to non-overlapping neighborhoods from an $N \times N \times C$ image yields a representation of size $N/2 \times N/2 \times 4 \cdot C$, thus containing four subsampled images each of size $N/2 \times N/2 \times C$. This can be repeated and allows for a partial autoregressive structure, where different subsampled images are generated sequentially and the pixels within the subsampled image can be generated in parallel.

The long-range sequential dependencies between the subsampled images are modeled through the internal state of the convolutional LSTMs [40]. Additionally, spatial dependencies within the subsampled images are captured by stacking multiple Conv-LSTM layers, providing a wide receptive field (see Sec. 4.1 for details). Following the Conv-LSTMs, unsqueeze operations bring the spatial resolution back to the size of the fine component, yielding the per-pixel parameters of the mixture distribution from Eq. (7).

Sampling. Fully autoregressive approaches require $\mathcal{O}(N_0^2)$ sampling steps. In contrast, our PixelPyramids – given sufficient parallel resources to sample a channel in parallel – require only a logarithmic number of steps:

Lemma 3.1. *Let the sampled image $\mathbf{I}$ be of resolution $[N_0, N_0, C]$, then the worst-case number of steps T (length of the critical path) required is $\mathcal{O}(\log N_0)$.*
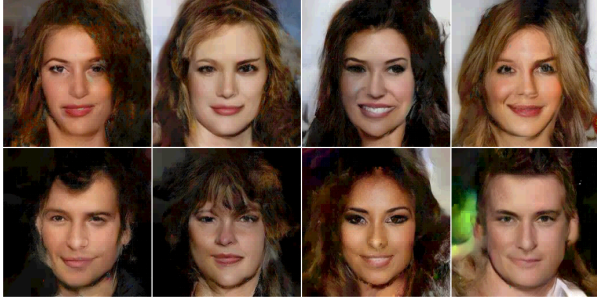
Proof. See supplemental material.

Figure 5. Random samples from 5-bit CelebA-HQ (256×256).

Figure 6. Random samples from 8-bit CelebA-HQ (1024×1024).

| Method | CelebA-HQ | LSUN | | |
		Bedroom	Tower	Church
SPN [32]	**0.61**	–	–	–
Glow [25]	1.03	1.20	–	–
MaCoW [31]	0.67	0.98	1.02	1.09
WaveletFlow [50]	0.94	–	–	–
PixelPyramids (*ours*)	**0.61**	**0.88**	**0.95**	**1.07**

Table 1. Evaluation on the 5-bit CelebA-HQ (256×256) and LSUN (128×128) datasets in bits/dim (lower is better).

Method	bits/dim. ($\downarrow$)
ParallelWavelet [25]	3.55
SPN [32]	**3.08**
PixelPyramids (*ours*)	3.40

Table 2. Evaluation on 8-bit ImageNet (128×128) dataset.

Method	bits/dim. ($\downarrow$)
WaveletFlow [50]	1.34
PixelPyramids (*ours*)	**0.58**

Table 3. Evaluation on 8-bit CelebA-HQ (1024×1024).

4. Experiments

To show the advantages of our PixelPyramids for density estimation and high-resolution image synthesis, we evaluate on the CelebA-HQ (256×256, 1024×1024) [23], LSUN – bedroom, church, and tower (128×128) [49], and ImageNet (128×128) [38] datasets. We report the density estimates as per-pixel likelihoods in terms of bits per dimension [46]. Additionally, we provide details on the computational efficiency of our model in comparison to competing methods in terms of training and sampling speed.

4.1. Implementation details

The number of pyramid levels (L) are chosen such that the coarsest scale has a spatial resolution of 4×4. Thus, $L = 16$ for CelebA-HQ (1024×1024). The U-Net applied to the fine components consists of two downsampling and two upsampling layers at level $L - 1$; an additional down/upsampling layer is added as we move to higher resolutions in the pyramid. Two squeeze operations were found to be sufficient to model local dependencies at levels $\{1, \ldots, L - 1\}$ in all experiments (see Tab. 4). The ConvLSTM consists of 2 layers with 128 channels and 3×3 filters. The autoregressive model at the coarsest level is a PixelCNN++ [39] with 96 channels and 5 residual layers. The number of mixture components across datasets is set to 10. The model is trained with the Adam optimizer [24] with an initial learning rate of 10^{-3}. The batch size is set to 64 for all datasets except for CelebA-HQ 1024×1024, where it is 32. For the high-resolution setting, *i.e.* 1024×1024, we use patchwise training [50]. This is enabled by our fully convolutional model (see supplemental material). Furthermore,

the per-pixel values of the bimodal distribution of the fine components (*cf*. Fig. 4) with its cyclic structure are shifted by 128 before encoding with Eq. (7), resulting in a unimodal distribution and providing for a better model fit.

4.2. Density estimation

We compare our approach for high-resolution images against models with state-of-the-art performance on challenging high-resolution datasets, the autoregressive Subscale Pixel Networks (SPN) [32] and flow-based models Glow [25], Masked Convolutional Generative Flow (MaCoW) [31], and WaveletFlow [50]. The number of bits per color channel for each setting is chosen to be consistent with prior work [25, 32, 50]. For CelebA-HQ (256×256) and LSUN (bedroom, tower, and church outdoor; 128×128), the experiments are performed on 5-bit images.

As Tab. 1 shows, our PixelPyramids outperform the flow-based models applied directly in image space, Glow and MaCoW, *e.g.* on CelebA-HQ by $\sim 40.7\,\%$ and $\sim 8.9\,\%$, respectively. The improvement in density estimation is consistent across datasets and illustrates the high representational power of our framework over standard flow models. Moreover, our approach outperforms WaveletFlow with its scale-space representations applied to flow by $\sim 35.1\,\%$. This highlights the advantage of our easier-to-model low-entropy Paired Pyramid representation. Our PixelPyramids close the performance gap to fully autoregressive models like SPN despite their partial autoregressive structure.

Tab. 2 further compares our approach on ImageNet 128×128 (8 bits). Our block-autoregressive PixelPyramids yield

(a) Random samples from 5-bit LSUN Bedroom (128 × 128).

(b) Random samples from 5-bit LSUN tower (128 × 128).

(c) Random samples from 5-bit LSUN church outdoor (128 × 128).

(d) Random samples from 8-bit ImageNet (128 × 128).

Figure 7. Random samples from our *PixelPyramids* model on different datasets.

better density estimates compared to the multi-scale ParallelWavelet, which applies a PixelCNN at every pyramid level to the sub-images in pixel space. This shows the advantage of our Paired Pyramid representation, including that it can be modeled well with only a partially autoregressive structure of the Conv-LSTM (see also Tab. 4). While SPNs still show better density estimates for this dataset, they are fully autoregressive in the spatial dimensions as well as for the image values, making them not scalable.

One of the main advantages of our approach is its applicability to high-resolution images. WaveletFlow is the only previous work based on exact inference that reports density estimates on 8-bit images of CelebA-HQ (1024 × 1024). PixelPyramids yield *a new state of the art* where density estimates are improved to ∼44 % of the bits/dim obtained with WaveletFlow, *i.e.* from 1.34 bits/dim to 0.58 bits/dim (Tab. 3). This large gain can be attributed to the ease of modeling the fine components from the Paired Pyramid representation compared to the scale-specific representations from the Haar wavelet. Moreover, at every level the Conv-LSTMs model the local structure in the scale-specific representations of our approach compared to WaveletFlow, which builds upon non-autoregressive models like Glow.

Note that the Paired Pyramid and consequently PixelPyramids have limited applicability to low-resolution images, *e.g.* CIFAR10 [29], as they yield fine components with similar (high) entropy as the original image (see supp. mat.).

Ablation studies. To validate the advantage of encoding densities with multi-scale low-entropy modulo differences, we compare against a baseline variant without the

Method	bits/dim. (↓)
PixelPyramids	0.61
PixelPyramids (no modulo difference)	0.76
PixelPyramids (non-autoregressive, $n_{\text{squeeze}} = 0$)	0.83
PixelPyramids ($n_{\text{squeeze}} = 1$)	0.62
PixelPyramids ($n_{\text{squeeze}} = 2$)	0.61
PixelPyramids ($n_{\text{squeeze}} = 3$)	0.61

Table 4. Model ablations on CelebA (256 × 256).

modulo difference, where the joint pixel distribution is decomposed similar to a regular image pyramid as $p(\mathbf{I}_0) = p(\mathbf{I}_L) \prod_{i=1}^{L} p(\mathbf{I}'_i \mid \mathbf{I}_i)$; all other model details remain unchanged. We term this *PixelPyramids (no modulo difference)*. From Tab. 4, we observe a clear drop in performance from 0.61 bits/dim to 0.76 bits/dim, indicating that the images $\mathbf{I}'_i$ have complex structural dependencies, which are difficult to model with a computationally inexpensive partially autoregressive structure. This shows the benefit of the low-entropy modulo difference for generative modeling.

Furthermore, *PixelPyramids (non-autoregressive)* shows the importance of modeling local spatial dependencies at each level. To assess this, we use $n_{\text{squeeze}} = 0$ squeeze operations, which effectively disables the Convolutional LSTM. Without the partial autoregressive structure, the model cannot encode the local dependencies (*cf.* Fig. 1), leading to a clear performance drop from 0.61 bits/dim to 0.83 bits/dim. This highlights that the combination of a low-entropy representation and a partial autoregressive structure is an important design choice for effective density estimation.

Finally, we analyze the effect of varying the number of squeeze operations, n_{squeeze}, which essentially regulates the
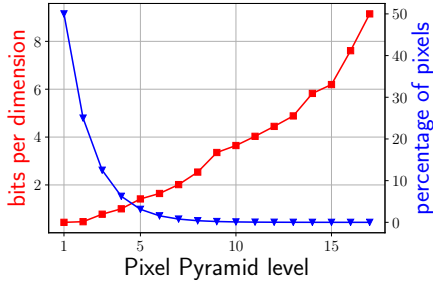
Figure 8. *Contribution of each pyramid level* to density estimates under the model *(red)* and the percentage of pixels *(blue)*.

sequential dependencies modeled by the Conv-LSTM in our framework. We observe that the density estimates improve significantly from 0.83 bits/dim to 0.62 bits/dim with a single squeeze operation, and improve further by 0.01 bits/dim with the default of two squeeze operations. $n_{squeeze} = 3$ leads to no additional benefit, since $n_{squeeze} = 2$ already implies a fully-autoregressive modeling at higher pyramid levels with limited spatial resolution, which have the most complex dependency structure (*cf.* Figs. 4 and 8). Furthermore, the higher resolution levels containing the vast majority of pixels have low density estimates under the model with $n_{squeeze} = 2$ compared to pixels at lower resolution levels (Fig. 8). This demonstrates the sufficiency of modeling local dependencies at higher resolutions in PixelPyramids.

Image synthesis. Qualitative image synthesis results on the 5-bit CelebA (256×256, Fig. 5), 8-bit CelebA (1024×1024, Fig. 6), 5-bit LSUN (bedroom, tower, and church outdoor; 128×128), as well as 8-bit ImageNet (128×128) datasets (Fig. 7) show the ability of our PixelPyramids to generate high-quality, diverse samples of high resolution. Since the sampling step involves the inverse transformation of Eq. (3), cyclic shifts of pixel values can occur if sampled values fall in the vicinity of 0 or 255. This leads to occasional pixel-level outliers (also noticeable in Fig. 6 as red salt & pepper noise). This could be resolved by replacing outlier pixels with the median of their neighborhood. We provide additional analyses in the supplemental material.

Computational cost. The computational efficiency of our model is shown in Tab. 5. All the experiments are performed on a single Nvidia TITAN X GPU. The number of parameters of PixelPyramids is similar to that of Glow and MaCoW. Despite having more parameters than WaveletFlow, the training speed of our model is of the same order, owing to its fully convolutional architecture at every level. Our block-autoregressive structure provides efficient sampling speeds compared *even* to flow-based models, which already allow for fast sampling. Moreover, our sampling speeds are *orders of magnitude* faster than fully autoregressive models. Thus, PixelPyramids provide both a computationally efficient setup *and* accurate density estimation.
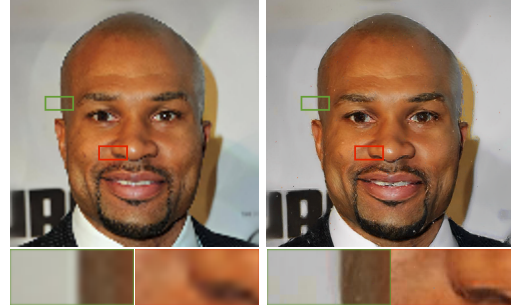

Figure 9. Super-resolution with PixelPyramids resizing a 128×128 image to 1024×1024 on the 8-bit CelebA-HQ (1024×1024).

Method	Params	Training Speed		Sampling Speed	
PixelCNN++ [39]	70 M	1.35	$\pm 0.10 \ \times 10^2$	6.00	$\pm 0.02 \ \times 10^6$
SPN[3] [32]	50 M	–		–	
Glow [25]	171 M	1.79	$\pm 0.10 \ \times 10^3$	5.31	$\pm 0.34 \ \times 10^2$
MaCoW [31]	177 M	–		4.34	$\pm 0.20 \ \times 10^2$
WaveletFlow [50]	52 M	1.47	$\pm 0.20 \ \times 10^1$	9.98	$\pm 0.86 \ \times 10^2$
PixelPyramids	166 M	1.98	$\pm 0.16 \ \times 10^1$	7.00	$\pm 0.10 \ \times 10^1$

Table 5. Comparison of training and sampling speeds (ms/image) on CelebA-HQ (256×256) with 100 random seeds.

Super-resolution. Similar to WaveletFlow [50], we show qualitatively the applicability of PixelPyramids for super-resolution without any explicit task-specific learning (Fig. 9). Given the low-resolution coarse image, the fine components are iteratively generated to obtain the high-resolution output. The integration of PixelPyramids with methods for probabilistic super-resolution [30] is an interesting direction of future research.

5. Conclusion

We present PixelPyramids, a novel and computationally efficient block-autoregressive framework for estimating densities and synthesis of high-resolution images. We show that multi-scale lossless representations obtained from Paired Pyramids, consisting of low-entropy fine components, provide for an easier to model localized dependency structure. This makes exact inference computationally efficient through partial autoregressive connections. Our framework provides a promising direction for extending exact inference models to high-resolution datasets.

---

[3]The sampling speed of SPN could not be reported due to the non-availability of an official implementation. Therefore, the sampling speed of PixelCNN++ with 0.67 bits/dim and same order of sampling speed (our implementation reproducing results on CIFAR10) is included as reference.

References

[1] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *arXiv:1907.02392 [cs.CV]*, 2019. 2

[2] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. Density modeling of images using a generalized normalization transformation. In *ICLR*, 2016. 1

[3] Apratim Bhattacharyya, Shweta Mahajan, Mario Fritz, Bernt Schiele, and Stefan Roth. Normalizing flows with multi-scale autoregressive priors. In *CVPR*, pages 8412–8421, 2020. 1

[4] Apratim Bhattacharyya, Christoph-Nikolas Straehle, Mario Fritz, and Bernt Schiele. Haar wavelet based block autoregressive flows for trajectories. In *GCPR*, pages 275–288, 2020. 3

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019. 1

[6] Joan Bruna, Pablo Sprechmann, and Yann LeCun. Super-resolution with deep convolutional sufficient statistics. In *ICLR*, 2016. 1

[7] Peter J. Burt and Edward H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Trans. Commun.*, 31(4):532–540, 1983. 2, 3

[8] Sheng Cao, Chao-Yuan Wu, and Philipp Krähenbühl. Lossless image compression through super-resolution. *arXiv:2004.02872 [eess.IV]*, 2020. 1

[9] Xi Chen, Nikhil Mishra, Mostafa Rohaninejad, and Pieter Abbeel. Pixelsnail: An improved autoregressive generative model. In *ICML*, pages 864–872, 2018. 2

[10] Emily L. Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep generative image models using a Laplacian pyramid of adversarial networks. In *NIPS*, pages 1486–1494, 2015. 2

[11] Sander Dieleman, Aäron van den Oord, and Karen Simonyan. The challenge of realistic music generation: Modelling raw audio at scale. In *NeurIPS*, pages 7989–7999, 2018. 1

[12] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. In *ICLR Workshop*, 2015. 1, 2, 5

[13] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *ICLR*, 2017. 1, 2

[14] Justin Domke, Alap Karapurkar, and Yiannis Aloimonos. Who killed the directed model? In *CVPR*, 2008. 1, 2

[15] Garoe Dorta, Sara Vicente, Lourdes Agapito, Neill D. F. Campbell, Simon Prince, and Ivor Simpson. Laplacian pyramid of conditional variational autoencoders. In *CVMP*, number 7, pages 1–9, 2017. 2

[16] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014. 1

[17] Alex Graves. Generating sequences with recurrent neural networks. *arXiv:1308.0850 [cs.NE]*, 2013. 2

[18] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *ICML*, pages 2722–2730, 2019. 1, 2

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 2

[20] Emiel Hoogeboom, Rianne van den Berg, and Max Welling. Emerging convolutions for generative normalizing flows. In *ICML*, pages 2771–2780, 2019. 2

[21] Wei-Ning Hsu, Yu Zhang, Ron J. Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, Patrick Nguyen, and Ruoming Pang. Hierarchical generative modeling for controllable speech synthesis. In *ICLR*, 2019. 1

[22] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aäron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. In *ICML*, pages 2410–2419, 2018. 2

[23] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 1, 2, 6

[24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *ICLR*, 2015. 6

[25] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, pages 10215–10224, 2018. 1, 2, 6, 8

[26] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *NIPS*, pages 4743–4751, 2016. 2

[27] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *ICLR*, 2014. 1

[28] Alexander Kolesnikov and Christoph H. Lampert. PixelCNN models with auxiliary variables for natural image modeling. In *ICML*, pages 1905–1914, 2017. 3

[29] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, U. of Toronto, 2009. 7

[30] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. SRFlow: Learning the super-resolution space with normalizing flow. In *ECCV*, volume 5, pages 715–732, 2020. 1, 8

[31] Xuezhe Ma, Xiang Kong, Shanghang Zhang, and Eduard H. Hovy. MaCow: Masked convolutional generative flow. In *NeurIPS*, pages 5891–5900, 2019. 1, 2, 6, 8

[32] Jacob Menick and Nal Kalchbrenner. Generating high fidelity images with subscale pixel networks and multidimensional upscaling. In *ICLR*, 2019. 1, 2, 5, 6, 8

[33] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, pages 4055–4064, 2018. 2

[34] Ali Razavi, Aäron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *NeurIPS*, pages 14837–14847, 2019. 1, 2

[35] Scott E. Reed, Aäron van den Oord, Nal Kalchbrenner, Sergio Gomez Colmenarejo, Ziyu Wang, Yutian Chen, Dan Belov, and Nando de Freitas. Parallel multiscale autoregressive density estimation. In *ICML*, pages 2912–2921, 2017. 1, 2

[36] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, pages 1278–1286, 2014. 1

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, volume 3, pages 234–241, 2015. 2, 4, 5

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(13):211–252, 2015. 2, 6

[39] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. PixelCNN++: Improving the PixelCNN with discretized logistic mixture likelihood and other modifications. In *ICLR*, 2017. 2, 5, 6, 8

[40] Xingjian Shi, Zhihan Gao, Leonard Lausen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Deep learning for precipitation nowcasting: A benchmark and a new model. In *NIPS*, pages 5617–5627, 2017. 2, 4, 5

[41] Yang Song, Chenlin Meng, Renjie Liao, and Stefano Ermon. Accelerating feedforward computation via parallel nonlinear equation solving. In *ICML*, pages 9791–9800, 2021. 2

[42] Lucas Theis and Matthias Bethge. Generative image modeling using spatial LSTMs. In *NIPS*, pages 1927–1935, 2015. 1

[43] H. H. Torbey and H. E. Meadows. System for lossless digital image coding/decoding. In *Visual Communications and Image Processing IV*, volume 1199, pages 989–1002. International Society for Optics and Photonics, SPIE, 1989. 1, 2, 3, 4

[44] Michael Tschannen, Eirikur Agustsson, and Mario Lucic. Deep generative models for distribution-preserving lossy compression. In *NeurIPS*, pages 5933–5944, 2018. 1

[45] Aäron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional image generation with PixelCNN decoders. In *NIPS*, pages 4790–4798, 2016. 1, 2, 5

[46] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *ICML*, pages 1747–1756, 2016. 1, 2, 6

[47] Aäron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C. Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, and Demis Hassabis. Parallel WaveNet: Fast high-fidelity speech synthesis. In *ICML*, pages 3918–3926, 2018. 1

[48] Auke J. Wiggers and Emiel Hoogeboom. Predictive sampling with forecasting autoregressive models. In *ICML*, pages 10260–10269, 2020. 2

[49] 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. *arXiv:1506.03365 [cs.CV]*, 2015. 2, 6

[50] Jason J. Yu, Konstantinos G. Derpanis, and Marcus A. Brubaker. Wavelet Flow: Fast training of high resolution normalizing flows. In *NeurIPS*, pages 6184–6196, 2020. 1, 3, 6, 8