Normalizing Flows with Multi-Scale Autoregressive Priors
– Supplemental Material –

Apratim Bhattacharyya[*1] Shweta Mahajan[*2] Mario Fritz[3] Bernt Schiele[1] Stefan Roth[2]

[1]Max Planck Institute for Informatics, Saarland Informatics Campus
[2]Department of Computer Science, TU Darmstadt
[3]CISPA Helmholtz Center for Information Security, Saarland Informatics Campus

We provide additional details of Lemma 4.1 in the main paper, additional details of our *mAR-SCF* model architecture, as well as additional results and qualitative examples.

A. Channel Dimensions in *mAR-SCF*

We begin by providing additional details of Lemma 4.1 in the main paper. We formally prove the claim that the number of channels at the last layer n (which does not have a SPLIT operation) is $\mathrm{C}_n = 2^{n+1} \cdot C$ for an image of size $[C, N, N]$. Thereby, we also show that the number of channels at any layer i (with a SPLIT operation) is $\mathrm{C}_i = 2^i \cdot C$.

Note that the number of time steps required for sampling depends on the number of channels for *mAR-SCF* (flow model) based on *MARPS* (Algorithm 1).

Lemma A.1. *Let the size of the sampled image be* $[C, N, N]$. *The number of channels at the last layer* $\mathbf{h}_n$ *is* $\mathrm{C}_n = 2^{n+1} \cdot C$.

Proof. The shape of image to be sampled is $[C, N, N]$. Since the network is invertible, the size of the image at level $\mathbf{h}_0$ is the size of the image sampled, *i.e.* $[C, N, N]$.

First, let the number of layers n be 1. This implies that during the forward pass the network applies a SQUEEZE operation, which reshapes the image to $[4C, N/2, N/2]$ followed by STEPOFFLOW, which does not modify the shape of the input, *i.e.* the shape remains $[4C, N/2, N/2]$. Thus the number of channels at the last layer $\mathbf{h}_1$ is $\mathrm{C}_1 = 4 \cdot C = 2^{1+1} \cdot C$ when $n = 1$.

Next, let us assume that the number of channels at the last layer $\mathbf{h}_{k-1}$ for a flow with $k-1$ layers is

$$\mathrm{C}_{k-1} = 2^k \cdot C. \tag{12}$$

We aim to prove by induction that the number of channels at the last layer $\mathbf{h}_k$ for a flow with k layers then is

$$\mathrm{C}_k = 2^{k+1} \cdot C. \tag{13}$$

To that end, we note that the dimensionality of the output at layer $k-1$ after SQUEEZE and STEPOFFLOW from Eq. (12) by assumption is $\mathrm{C}_{k-1} = 2^k \cdot C$. For a flow with k layers, the $(k-1)^{\text{st}}$ layer has a SPLIT operation resulting in $\{\mathbf{l}_{k-1}, \mathbf{r}_{k-1}\}$, each with size $[2^{k-1}C, N/2^{k-1}, N/2^{k-1}]$. The number of channels for $\mathbf{r}_{k-1}$ at layer $k-1$ is thus $2^k \cdot C/2 = 2^{k-1} \cdot C$. At layer k the input with $2^{k-1} \cdot C$ channels is transformed by SQUEEZE to $2^{k-1} \cdot 4 \cdot C = 2^{(k-1)+2} \cdot C = 2^{k+1} \cdot C$ channels.

Therefore, by induction Eq. (13) holds for $k = n$. Thus the number of channels at the last layer $\mathbf{h}_n$ is given as $\mathrm{C}_n = 2^{n+1} \cdot C$. □

B. *mAR-SCF* Architecture

In Fig. 6, we show the architecture of our *mAR* prior in detail for a layer i of *mAR-SCF*. The network is a convolutional LSTM with three LSTM layers. The input at each time-step is the previous channel of $\mathbf{l}_i$, concatenated with the output after convolution on $\mathbf{r}_i$. Our *mAR* prior autoregressivly outputs the probability of each channel $\mathbf{l}_i^j$ given by the distribution $p_\phi\left(\mathbf{l}_i^j | \mathbf{l}_i^{j-1}, \cdots, \mathbf{l}_i^1, \mathbf{r}_i\right)$ modeled as $\mathcal{N}\left(\mu_i^j, \sigma_i^j\right)$ during inference. Because of the internal state of the Convolutional LSTM (memory), our *mAR* prior can learn long-range dependencies.

In Fig. 2 in the main paper, the STEPOFFLOW operation consists of an activation normalization layer, an invertible 1×1 convolution layer followed by split coupling layers (32 layers for affine couplings or 4 layers of MixLogCDF at each level).

C. Empirical Analysis of Sampling Speed

In Fig. 7, we analyze the real-world sampling time of our state-of-the-art *mAR-SCF* model with MixLogCDF couplings using varying image sizes $[C, N, N]$. In particular, we vary the input spatial resolution N. We report the mean and variance of 1000 runs with batch size of 8 on an Nvidia V100 GPU with 32GB memory. Using smaller batch sizes

[*]Authors contributed equally

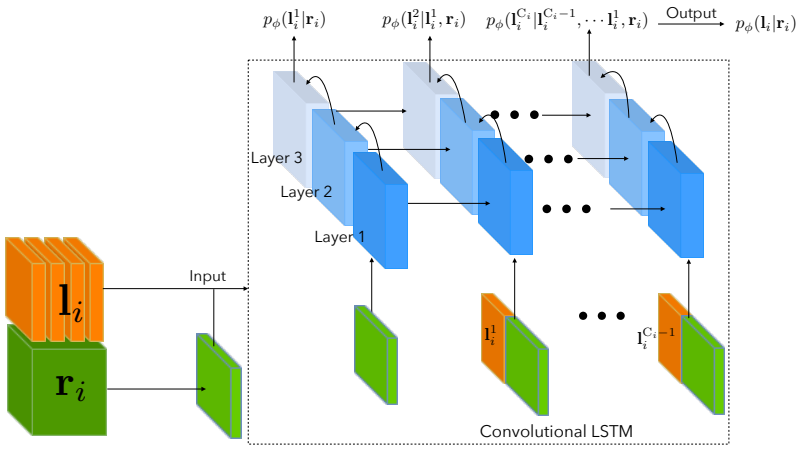


Figure 6. Architecture of our multi-scale autoregressive (*mAR*) prior.

of 8 ensures that we always have enough parallel resources for all image sizes. Under these conditions, we see that the sampling time increases linearly with the image size. This is because the number of sampling steps of our *mAR-SCF* model scales as $\mathcal{O}(N)$, *cf*. Lemma 4.1.

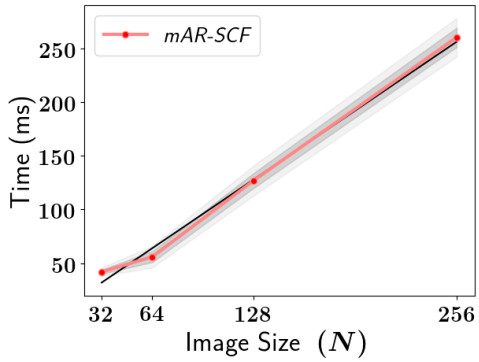


Figure 7. Analysis of real-world sampling time of our *mAR-SCF* model with varying image size $[C, N, N]$. For reference, we show the line $y = 2x$ in black.

D. Additional Qualitative Examples

We include additional qualitative examples for training our generative model on the MNIST, CIFAR10, and ImageNet (32×32) datasets. In Fig. 8, we see that the visual sample quality of our *mAR-SCF* model with MixLogCDF couplings is competitive with those of the state-of-the-art Residual Flows [4]. Moreover, note that we achieve better test log-likelihoods (0.88 *vs*. 1.00 bits/dim).

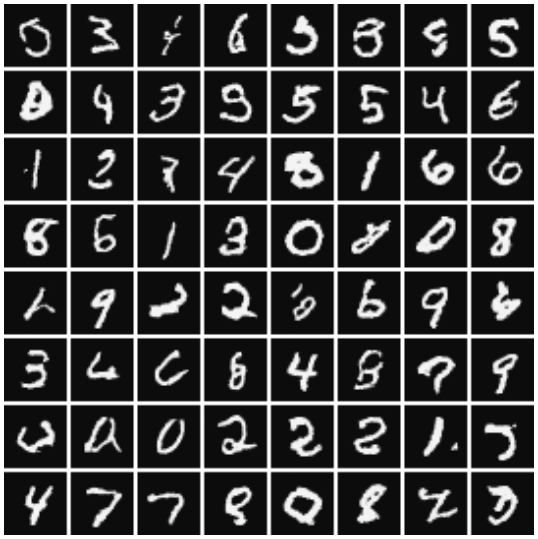
We additionally provide qualitative examples for ImageNet (32×32) in Figure 10. We observe that in comparison to the state-of-the-art Flow++ [14] and Residual Flows [4], the images generated by our *mAR-SCF* model are detailed with a competitive visual quality.

Finally, on CIFAR10 we compare with the fully autoregressive PixelCNN model [38] in Figure 9. We see that although the fully autoregressive PixelCNN achieves significantly better test log-likelihoods (3.00 *vs*. 3.24 bits/dim), our *mAR-SCF* models achieves better visual sample quality (also shown by the FID and Inception metrics in Table 3 of the main paper).

E. Additional Interpolation Results

Finally, in Fig. 11 we show qualitative examples to compare the effect using our proposed interpolation method (Eq. 11) versus simple linear interpolation in the multimodal latent space of our *mAR* prior. We use the Adamax optimizer to optimize Eq. (11) with a learning rate of 5×10^{-2} for $\sim$ 100 iterations. The computational requirement per iteration is approximately equivalent to a standard training iteration of our *mAR-SCF* model, *i.e*. $\sim$ 1 sec for a batch of size 128. We observe that interpolated images obtained from our *mAR-SCF* affine model using our proposed interpolation method (Eq. 11) have a better visual quality, especially for the interpolations in the middle of the interpolating path. Note that we find our scheme to be stable in practice in a reasonably wide range of the hyperparameters $\lambda_2 \approx \lambda_1 \in [0.2, 0.5]$ as it interpolates in high-density regions between $\mathbf{x}_\text{A}, \mathbf{x}_\text{B}$. We support the better visual quality of the interpolations of our scheme compared to the linear method with Inception scores computed for interpolations obtained from both methods in Fig. 12.

(a) Residual Flows [4]



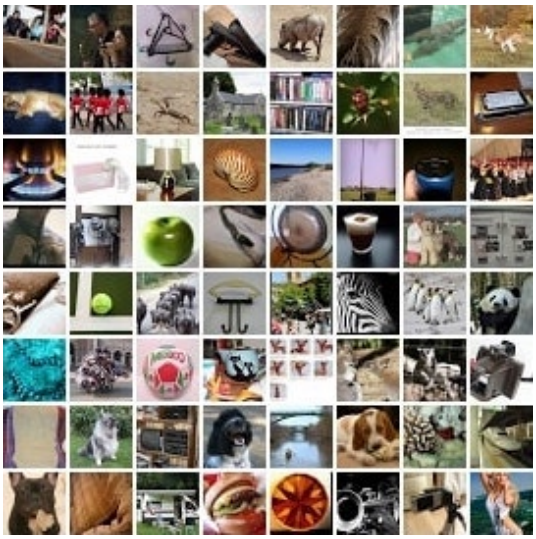
(b) Our *mAR-SCF* (MixLogCDF)

Figure 8. Random samples when trained on MNIST.

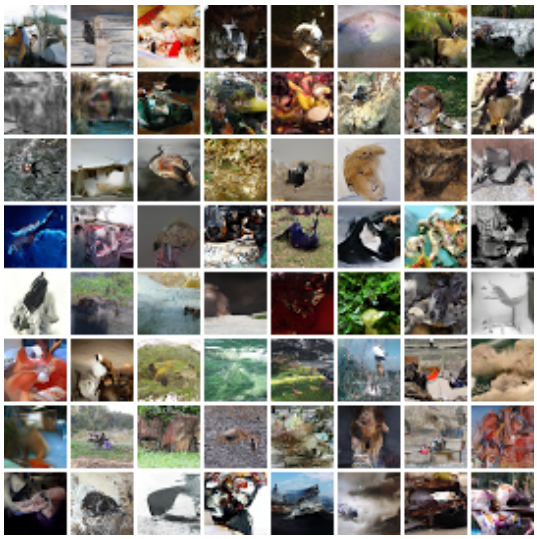
(a) PixelCNN [38]

(b) Our *mAR-SCF* (MixLogCDF)

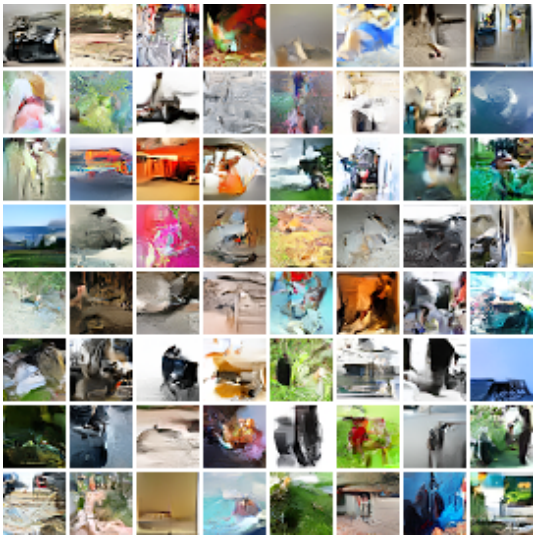
Figure 9. Random samples when trained on CIFAR10 (32×32).



(a) Real Data



(b) Flow++ [14]



(c) Residual Flows [4]

(d) Our *mAR-SCF* (MixLogCDF)

Figure 10. Random samples when trained on ImageNet (32×32).

Figure 11. Comparison of interpolation quality of our interpolation scheme (Eq. 11) with a standard linear interpolation scheme. The top row of each result shows the interpolations between real images for the linear method and the corresponding bottom rows are the interpolations with our proposed scheme.

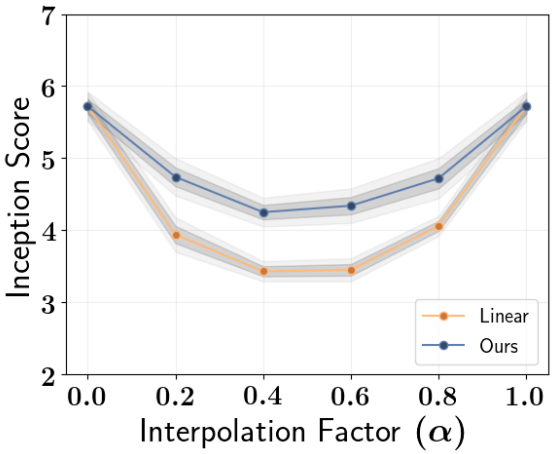


Figure 12. Inception scores of random samples generated with our interpolation scheme versus linear interpolation.