Generic Image Restoration with Flow Based Priors

Leonhard Helminger[1,*] Michael Bernasconi[1,*] Abdelaziz Djelouah[2]
Markus Gross[1] Christopher Schroers[2]

[1]Department of Computer Science
ETH Zurich, Switzerland

[2]DisneyResearch|Studios
Zurich, Switzerland

Figure 1: Supplementary comparison with DIP (1) on JPG artifact removal. We can observe that our method achieves competitive results, with for example sharper lines around the eyes.

A. Summary of Changes

The CVPR reviewers appreciated the potential of the work despite the experimental sections that did not consider a sufficiently large variety of works in the comparisons. Our submission to NTIRE2021 doesn't include changes as we believe the workshop is the ideal venue to discuss such ideas at an early stage. We would like to emphasize that Normalizing Flows (NF) exhibit advantageous properties and there is a growing interest in extending their usage. We are aware that our model has limitations compared to approaches leveraging more established models such as GANs. However, to demonstrate that research in this new direction is interesting even in its early stage, we think our focused comparison to "Deep Image Prior" as a baseline is justified.

B. Additional Comparison with DIP (1)

We provide an additional comparison with Deep Image Prior for the task of compression artifact removal. We can observe that our method achieves competitive results, with for example sharper lines around the eyes (Figure 1).

C. MNIST

For MNIST the network architecture is kept simple, only consisting of a single level. We use $K = 16$ steps in our model. As coupling transform we use the one depicted in Figure 2 with two blocks ($N = 2$) and 128 intermediate channels ($C_{inter} = 128$). Finally, we choose a Gaussian with unit variance as our base distribution. The Gaussian's mean is set to a trainable parameter. All other parameters are listed in Table 1.

Parameter	**Value**
# levels	1
# flow blocks per level N_f	16
Affine coupling C_{inter}	128
Base distribution $p(\mathbf{u}_0)$	$\mathcal{N}(\mu, 1)$
Optimizer	Adam
learning rate	10^{-4}
batch size	50
# steps	10^5
max gradient value	10^5
max gradient L_2-norm	10^4

Table 1: Details of architecture and training for the MNIST experiments

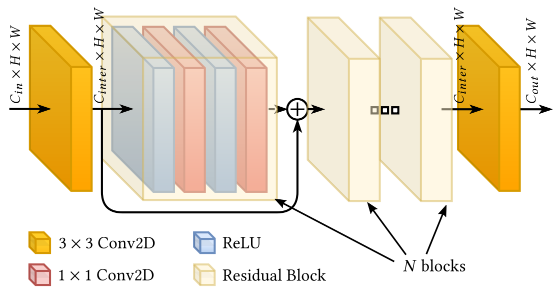

Figure 2: Details of the affine coupling transform. 3×3 Conv2D and 1×1 Conv2D refer to standard 2D convolutions using a kernel size of 3x3 and 1x1 respectively. The " $+$ " at the end of the block is an element wise addition.

D. Sprites

Each image in the Sprites dataset consists of a figure performing some pose in front of a random background. Figures are centered in the image and have varying color for hair and clothing. Each image is of size 64x64. Dataset will be made available upon acceptance.

Architecture. For this experiment, the number of levels is set to $L = 3$ and each level has $K = 8$ steps. The distributions $p(\mathbf{u}_1|\mathbf{h}_1)$ and $p(\mathbf{u}_2|\mathbf{h}_2)$ depend on a function which computes mean $\mu(\mathbf{h}_l)$ and variance $\sigma(\mathbf{h}_l)$. We call this function the context encoder. A single 2D convolution with kernel size 3×3 and twice the number of output dimension as input dimensions is used as the context encoder. The context encoder's output is then split in half along the channel dimension. One half is used as $\mu(\mathbf{h}_l)$, the other as $\sigma(\mathbf{h}_l)$. The convolutions weight and bias are initialized to zero for stability reasons. The other parameters for the Sprites dataset are listed in Table 2.

Parameter	Value
# levels (L)	3
# flow steps per level (K)	8
Affine coupling C_{inter}	128
Base distribution $p(\mathbf{u}_1\|\mathbf{h}_1)$, $p(\mathbf{u}_2\|\mathbf{h}_2)$	$\mathcal{N}(\mu(\mathbf{h}_l),\ \sigma(\mathbf{h}_l))$
Base distribution $p(\mathbf{u}_0)$	$\mathcal{N}(\mu,\ \sigma)$
Context Encoder $p(\mathbf{u}_1\|\mathbf{h}_1)$, $p(\mathbf{u}_2\|\mathbf{h}_2)$	Conv2D (zero init), kernel size 3x3
Optimizer	Adam
learning rate	10^{-4}
batch size	20
# steps	10^5
max gradient value	10^5
max gradient L_2-norm	10^4
latent noise magnitude	± 0.5
latent noise loss (β_{ln})	100
autoencoder loss (β_{ae})	1

Table 2: Sprites training specification.

E. DIV2K

The number of levels in the architecture is set to $L = 3$ with $K = 4$ steps per level. The number of intermediate channels in the coupling transforms is 256. The context encoder architecture is deepened from 1 to 5 convolutional layer as is illustrated in Figure 3 and a dropout layer is added to the beginning. All the architecture parameters are listed in Table 3.

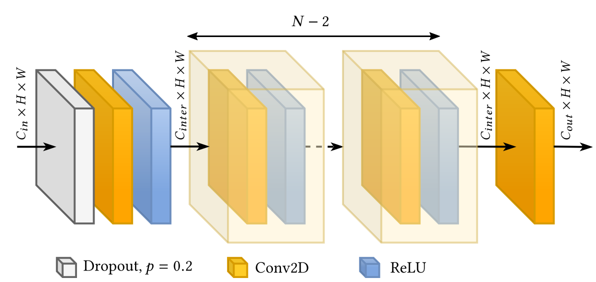

Figure 3: Architecture of the context encoder used for the DIV2K example. A dropout layer with $p = 0.2$ is used as the first layer to prevent overfitting. The last convolution's weight and bias are initialized to zero for stability reasons.

Parameter	Value
# levels (L)	3
# flow steps per level (K)	4
Affine coupling C_{inter}	256
Base distribution $p(\mathbf{u}_1\|\mathbf{h}_1)$, $p(\mathbf{u}_2\|\mathbf{h}_2)$	$\mathcal{N}(\mu(\mathbf{h}_l),\ \sigma(\mathbf{h}_l))$
Base distribution $p(\mathbf{u}_0)$	$\mathcal{N}(\mu,\ \sigma)$
Context Encoder $p(\mathbf{u}_1\|\mathbf{h}_1)$, $p(\mathbf{u}_2\|\mathbf{h}_2)$	$N = 5$
Optimizer	Adam
learning rate	10^{-4}
batch size	15
# steps	20^5
max gradient value	10^5
max gradient L_2-norm	10^4
latent noise magnitude	± 0.5
latent noise loss (β_{ln})	100
autoencoder loss (β_{ae})	1
image noise loss (β_{in})	100
image noise magnitude	± 10

Table 3: DIV2K training specification.

Patch-wise Reconstruction. A full image of arbitrary size can be reconstructed by reconstructing each patch individually. To avoid boundary artifacts between patches a margin is used as illustrated in Figure 4. The margin causes overlap between adjacent patches yielding more consistent results in boundary regions.

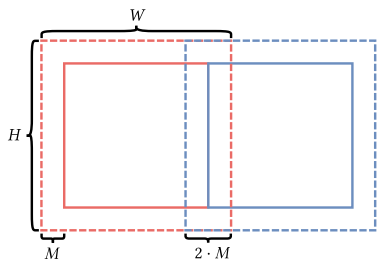

Figure 4: Illustrations of the tiles used for patch-wise reconstruction. H and W refer to the patches height and with respectively. M refers to the margin. Neighboring patches overlap in a region of width $2 \cdot M$. Analogously the same pattern extends in the vertical direction. In our work we use $H = W = 64$ and $M = 4$.

References

[1] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018. 1