Supplemental Material (Noise2Noise)

Jaakko Lehtinen[1 2] **Jacob Munkberg**[1] **Jon Hasselgren**[1] **Samuli Laine**[1] **Tero Karras**[1] **Miika Aittala**[3] **Timo Aila**[1]

1. Network architecture

Table 1 shows the structure of the U-network (Ronneberger et al., 2015) used in all of our tests, with the exception of the first test in Section 3.1 that used the "RED30" network (Mao et al., 2016). For all basic noise and text removal experiments with RGB images, the number of input and output channels were $n = m = 3$. For Monte Carlo denoising we had $n = 9, m = 3$, i.e., input contained RGB pixel color, RGB albedo, and a 3D normal vector per pixel. The MRI reconstruction was done with monochrome images ($n = m = 1$). Input images were represented in range $[-0.5, 0.5]$.

2. Training parameters

The network weights were initialized following He et al. (2015). No batch normalization, dropout or other regularization techniques were used. Training was done using ADAM (Kingma & Ba, 2015) with parameter values $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 10^{-8}$.

Learning rate was kept at a constant value during training except for a brief rampdown period at where it was smoothly brought to zero. Learning rate of 0.001 was used for all experiments except Monte Carlo denoising, where 0.0003 was found to provide better stability. Minibatch size of 4 was used in all experiments.

3. Finite corrupted data in L_2 minimization

Let us compute the expected error in L_2 norm minimization task when corrupted targets $\{\hat{y}_i\}_{i=1}^N$ are used in place of the clean targets $\{y_i\}_{i=1}^N$, with N a finite number. Let y_i be arbitrary random variables, such that $\mathbb{E}\{\hat{y}_i\} = y_i$. As usual, the point of least deviation is found at the respective mean. The expected squared difference between these means across realizations of the noise is then:

$$\begin{aligned}
&\mathbb{E}_{\hat{y}} \left[\frac{1}{N} \sum_i y_i - \frac{1}{N} \sum_i \hat{y}_i \right]^2 \\
=&\frac{1}{N^2} \left[\mathbb{E}_{\hat{y}} (\sum_i y_i)^2 - 2\mathbb{E}_{\hat{y}} \left[(\sum_i y_i)(\sum_i \hat{y}_i) \right] + \mathbb{E}_{\hat{y}} (\sum_i \hat{y}_i)^2 \right] \\
=&\frac{1}{N^2} \mathrm{Var}(\sum_i \hat{y}_i) \\
=&\frac{1}{N} \left[\frac{1}{N} \sum_i \sum_j \mathrm{Cov}(\hat{y}_i, \hat{y}_j) \right]
\end{aligned} \tag{1}$$

In the intermediate steps, we have used $\mathbb{E}_{\hat{y}}(\sum_i \hat{y}_i) = \sum_i y_i$ and basic properties of (co)variance. If the corruptions are mutually uncorrelated, the last row simplifies to

$$\frac{1}{N} \left[\frac{1}{N} \sum_i \mathrm{Var}(y_i) \right] \tag{2}$$

In either case, the variance of the estimate is the average (co)variance of the corruptions, divided by the number of samples N. Therefore, the error approaches zero as the number of samples grows. The estimate is unbiased in the sense that it is correct on expectation, even with a finite amount of data.

The above derivation assumes scalar target variables. When $\hat{y}_i$ are images, N is to be taken as the total number of scalars in the images, i.e., #images $\times$ #pixels/image $\times$ #color channels.

4. Mode seeking and the "L_0" norm

Interestingly, while the "L_0 norm" could intuitively be expected to converge to an exact mode, i.e. a local maximum of the probability density function of the data, theoretical analysis reveals that it recovers a slightly different point. While an actual mode is a zero-crossing of the *derivative* of the PDF, the L_0 norm minimization recovers a zero-crossing of its *Hilbert transform* instead. We have verified this behavior in a variety of numerical experiments, and, in practice, we find that the estimate is typically close to the true mode. This can be explained by the fact that the Hilbert transform

[*]Equal contribution [1]NVIDIA [2]Aalto University [3]MIT CSAIL. Correspondence to: Jaakko Lehtinen <jlehtinen@nvidia.com>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

NAME	N_{out}	FUNCTION
INPUT	n	
ENC_CONV0	48	Convolution 3×3
ENC_CONV1	48	Convolution 3×3
POOL1	48	Maxpool 2×2
ENC_CONV2	48	Convolution 3×3
POOL2	48	Maxpool 2×2
ENC_CONV3	48	Convolution 3×3
POOL3	48	Maxpool 2×2
ENC_CONV4	48	Convolution 3×3
POOL4	48	Maxpool 2×2
ENC_CONV5	48	Convolution 3×3
POOL5	48	Maxpool 2×2
ENC_CONV6	48	Convolution 3×3
UPSAMPLE5	48	Upsample 2×2
CONCAT5	96	Concatenate output of POOL4
DEC_CONV5A	96	Convolution 3×3
DEC_CONV5B	96	Convolution 3×3
UPSAMPLE4	96	Upsample 2×2
CONCAT4	144	Concatenate output of POOL3
DEC_CONV4A	96	Convolution 3×3
DEC_CONV4B	96	Convolution 3×3
UPSAMPLE3	96	Upsample 2×2
CONCAT3	144	Concatenate output of POOL2
DEC_CONV3A	96	Convolution 3×3
DEC_CONV3B	96	Convolution 3×3
UPSAMPLE2	96	Upsample 2×2
CONCAT2	144	Concatenate output of POOL1
DEC_CONV2A	96	Convolution 3×3
DEC_CONV2B	96	Convolution 3×3
UPSAMPLE1	96	Upsample 2×2
CONCAT1	$96+n$	Concatenate INPUT
DEC_CONV1A	64	Convolution 3×3
DEC_CONV1B	32	Convolution 3×3
DEV_CONV1C	m	Convolution 3×3, linear act.

Table 1. Network architecture used in our experiments. N_{out} denotes the number of output feature maps for each layer. Number of network input channels n and output channels m depend on the experiment. All convolutions use padding mode "same", and except for the last layer are followed by leaky ReLU activation function (Maas et al., 2013) with $\alpha = 0.1$. Other layers have linear activation. Upsampling is nearest-neighbor.

approximates differentiation (with a sign flip): the latter is a multiplication by $i\omega$ in the Fourier domain, whereas the Hilbert transform is a multiplication by $-i\,\mathrm{sgn}(\omega)$.

For a continuous data density $q(x)$, the norm minimization task for L_p amounts to finding a point x^* that has a minimal expected p-norm distance (suitably normalized, and omitting the pth root) from points $y \sim q(y)$:

$$\begin{aligned} x^* &= \operatorname*{argmin}_x \mathbb{E}_{y\sim q}\{\frac{1}{p}|x-y|^p\} \\ &= \operatorname*{argmin}_x \int \frac{1}{p}|x-y|^p q(y)\,\mathrm{d}y \end{aligned} \tag{3}$$

Following the typical procedure, the minimizer is found at a root of the derivative of the expression under argmin:

$$\begin{aligned} 0 &= \frac{\partial}{\partial x}\int \frac{1}{p}|x-y|^p q(y)\,\mathrm{d}y \\ &= \int \mathrm{sgn}(x-y)|x-y|^{p-1} q(y)\,\mathrm{d}y \end{aligned} \tag{4}$$

This equality holds also when we take $\lim_{p\to 0}$. The usual results for L_2 and L_1 norms can readily be derived from this form. For the L_0 case, we take $p = 0$ and obtain

$$\begin{aligned} 0 &= \int \mathrm{sgn}(x-y)|x-y|^{-1} q(y)\,\mathrm{d}y \\ &= \int \frac{1}{x-y} q(y)\,\mathrm{d}y. \end{aligned} \tag{5}$$

The right hand side is the formula for the Hilbert transform of $q(x)$, up to a constant multiplier.

References

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

Kingma, Diederik P. and Ba, Jimmy. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Maas, Andrew L, Hannun, Awni Y, and Ng, Andrew. Rectifier nonlinearities improve neural network acoustic models. In *Proc. International Conference on Machine Learning (ICML)*, volume 30, 2013.

Mao, Xiao-Jiao, Shen, Chunhua, and Yang, Yu-Bin. Image restoration using convolutional auto-encoders with symmetric skip connections. In *Proc. NIPS*, 2016.

Ronneberger, Olaf, Fischer, Philipp, and Brox, Thomas. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, 9351:234–241, 2015.