Supplementary Note

Alexander Krull\textsuperscript{1,2}, Tim-Oliver Buchholz\textsuperscript{2}, Florian Jug
\textsuperscript{1} krull@mpi-cbg.de
\textsuperscript{2} Authors contributed equally

MPI-CBG/PKS (CSBD), Dresden, Germany

This supplementary note provides additional information on \textsc{Noise2Void} and on the experiments we conducted. We will first discuss details on our training, validation and data preparation procedure (Section 1) and then continue to provide the size of our networks’ receptive fields (Section 2). Finally, we will provide additional results achieved by multiple variations of our method (Section 3).

In Supplementary Figures 1, 2, 3, and 4 we will show additional qualitative results.

1 Training, Validation and Data Preparation

We train all networks for 200 cycles, with each cycle consisting of 400 gradient steps. During the training of all networks we perform on-the-fly random sub-patch extraction (with a size of \(64 \times 64\)) from our previously prepared training data.

The validation loss, which is used by the CARE learning rate scheduler, is calculated like the training loss for traditional training, \textsc{N2N}, and \textsc{N2V} respectively. Using the ground truth for traditional training, and a second noisy image for \textsc{N2N}. For \textsc{N2V}, we calculate the validation loss using randomly selected masked pixels, like it is done for the training loss.

We will now go on to describe the training data preparation of the individual datasets and provide additional details.

1.1 BSD68 [2]

As in [4], we train our network using the 400 gray scale images with \(180 \times 180\) pixels. We add zero centered independent Gaussian noise (\(\sigma = 25\)) to all patches. Then we randomly select 1\% of all patches to serve as validation dataset. Finally, we augment our data by adding all rotated (multiples of \(90^\circ\)) and mirrored versions. During training with \textsc{N2V}, we use the Uniform Pixel Selection \textsc{UPS} masking method with a \(5 \times 5\) pixel window (see Section 3.1).

1.2 Simulated Microscopy

In the simulation of our microscopy data, we start out with three independently generated 3D phantom image stacks (size: \(256 \times 256 \times 256\)) that describe the fluorophore distribution in our sample. We use two of the phantoms to generate training data and one to generate test data. We start the microscope simulation by normalizing the phantoms to the range \([0, 1]\). We then add a constant value of 0.2 to simulate background illumination and multiply the phantoms by a factor \(\lambda\) to account for the exposure. Subsequently we simulate shot noise. Every pixel value is generated by drawing from a Poisson distribution using the shifted and scaled phantom as the expected value of the pixel. Finally, we add pixel-wise independent zero-mean Gaussian noise with a standard deviation of \(\sigma = 1\) to simulate readout noise in the sensor. According to the above scheme, we generate low- and high-exposure-images, using \(\lambda = 20\) and \(\lambda = 10000\) respectively. A second low-exposure-image (\(\lambda = 20\)) is generated for \textsc{N2N} training. The high-exposure-images serve as ground truth for the traditional training and are used to calculate the PSNR during testing. With respect to the latter, we account for different exposure rates by scaling our network output with the corresponding factor. Since \textsc{N2V} is currently only implemented in 2D, we view each of the generated 3D image stacks as collection of 2D images and process them independently. We use 10\% of out training images as validation dataset. During training with \textsc{N2V}, we use the Gaussian Pixel Selection (\textsc{GPS}) masking method (see Section 3.1).

1.3 Fluo-C2DL-MSC (CTC-MSC) [3]

We train our network using the two training movies provided in the cell tracking challenge\textsuperscript{1}. We use only the image data without any additional annotations like ground truth segmentations. This is the same data we use for testing as well. We iterate over all frames of both movies and randomly extract patches of \(80 \times 80\) pixels. To reduce the amount of patches showing only background we calculate the standard deviation of the intensities of each patch. We exclude all patches with standard deviation below 1250. For each frame we continue to sample until we find 256 patches that pass this test. We use one of these patches from every frame in our validation set. We augment our data by adding all rotated (multiples of \(90^\circ\)) and mirrored versions. During training with \textsc{N2V}, we use the \textsc{UPS} masking method with a \(5 \times 5\) pixel window (see Section 3.1).

\textsuperscript{1}http://celltrackingchallenge.net/
1.4 Fluo-N2DH-GOWT1 (CTC-N2DH) [3]

The Fluo-N2DH-GOWT1 training dataset also consists of two movies. Again we use only the image data without any additional annotations like ground truth segmentations. This is again the same data we use for testing as well. The data preparation is done as for the CTC-MSC dataset. However, here we exclude all patches with standard deviation below 5. During training with N2V, we use the UPS masking method with a 5 × 5 pixel window (see Section 3.1).

1.5 Cryo-TEM [1]

We use a single 7676 × 7420 pixel cryo-TEM image as source for our training data and extract 435 overlapping patches of size 512 × 512. The same image is later used for testing. We randomly select 10% of the patches and use them as validation dataset. Here, we do not use data augmentation in the form of flipping or rotating. During training with N2V, we use the UPS masking method with a 5 × 5 pixel window (see Section 3.1).

2 Receptive Fields of our Networks

We used the CARE framework to calculate the size of our networks’ receptive fields: A CARE network with kernel size of 3 × 3 and depth 2 has a receptive field of 22 × 22 pixels. A network with kernel size of 5 × 5 has a receptive field of size 40 × 40.

3 Variations of N2V and Parameters

We used the BSD68 dataset to test different parameter settings and to try different variants of our masking scheme. The results can be found in Supplementary Table 1.

3.1 Masking Methods

We tried the following variants of our masking scheme: Uniform Pixel Selection (UPS) replaces the value of the selected pixel with a randomly selected pixel value from a square window around $i$. This includes the pixel itself. In Supplementary Table 1 we write the size of the window in parenthesis.

Gaussian (G) masking changes the value of the selected pixel $i$ by adding random Gaussian noise. In Supplementary Table 1 we provide the standard deviation of the noise in parenthesis. The Gaussian Fitting (GF) method looks at a local square neighborhood of the pixel $i$ and fits a 1D Gaussian distribution to the pixel values by calculating mean and standard deviation. The pixel $i$ is included in this calculation. The pixel value is masked by drawing from the fitted Gaussian distribution. In Supplementary Table 1 we write the size of the neighborhood in parenthesis.

The Gaussian Pixel Selection (GPS) replaces the value of the selected pixel $i$ with a randomly selected pixel that is found by drawing from a 2D isotropic Gaussian distribution centered at pixel $i$. The pixel $i$ itself is excluded from this draw. That is, if the pixel $i$ is selected by the Gaussian draw, we redraw until another pixel has been selected. We use a Gaussian distribution with a standard deviation of $\sigma = 4$.

3.2 Other Parameters

In addition to the masking method, we also experimented with other parameters. In particular, we tried using the mean absolute error (MAE) as our loss function instead of the mean square error (MSE). We also varied the number of feature channels used in the first layer, denoted as Features in Supplementary Table 1, and the size of the convolutional kernels used in the network, denoted as Kernel.

References

Figure 1: Qualitative NOISE2VOID results on the BSD68 dataset.
Figure 2: Qualitative NOISE2VOID results from the cryo-TEM dataset.
Figure 3: Qualitative NOISE2VOID results from the CTC-MSC Dataset.
Figure 4: Qualitative NOISE2VOID results from the CTC-N2DH dataset.