Denoising diffusion models for high-resolution microscopy image restoration

Pamela Osuna-Vargas[1,2] Maren H. Wehrheim[1,2] Lucas Zinz[2] Johanna Rahm[3]
Ashwin Balakrishnan[3] Alexandra Kaminer[3] Mike Heilemann[3] Matthias Kaschube[1,2]
[1]Frankfurt Institute for Advanced Studies
[2]Department of Computer Science and Mathematics, Goethe University Frankfurt
[3]Institute of Physical and Theoretical Chemistry, Goethe University Frankfurt
{osuna, wehrheim, kaschube}@fias.uni-frankfurt.de, s7005665@stud.uni-frankfurt.de
{rahm, balakrishnan, kaminer, heilemann}@chemie.uni-frankfurt.de

S1. Datasets

S1.1. Data pre-processing

For the microtubules and synapse datasets, we used the itk library v5.4rc1 [6] to rigidly align the pairs of low- and high-resolution images. The registered image contains padded pixels, while the reference image does not. Thus, to avoid the models from learning misleading information, we used the resulting transformation to reproduce the padding in the reference image. For all datasets, images were cropped into patches of size 256 × 256 pixels in a non-overlapping-fashion.

S1.2. Dataset partitioning

We here describe (Table S1) the number of FOVs, image sizes and dataset partitions used for training, validation and during test time.

Dataset	FOVs	Orig. image size (px)	Train	Validation	Test
Microtubules	104	2560 × 2560	1272	89	265
Mitochondria	345	600 × 600	2646	153	306
Synapses	24	550 × 550 × 20	1198	56	112
Zebrafish	20	512 × 512	3600 (72)	200 (4)	200 (4)

Table S1. **Dataset description and pre-processing**. For each dataset, we report the number of fields of view (FOVs), the image sizes, as well as the number of FOVs for the train, validation and test partitions. For the zebrafish dataset, the same sample is consecutively captured 50 times, exhibiting different noise realizations. Thus, we report in parenthesis the number of different sub-FOVs (after dividing the original into patches) before using all noise realizations.

S1.3. Diversity

In Table S2 we highlight the dimensions along which the tested datasets vary.

#	Sample type	Imaging type	Condition	Raw	Ground Truth
1	Microtubules	STED	Fixed	Low-light dose	High-light dose
2	Mitochondria	STED	Living	Low-light dose	High-light dose
3	Synapses	Confocal	Fixed	Confocal	Super-resolution
4	Zebrafish	Confocal	Fixed	Single images	Avg. of 50 images

Table S2. **Differences between denoising datasets.** We test the denoising performance using four diverse datasets. These datasets vary along the sample and imaging type, the cell condition (fixed vs. live cells), as well as how the raw and ground truth data were generated.

S2. Additional quality control metrics

The mean absolute error (MAE) between the ground truth image y and reconstructed image $\hat{y}$ captures the general offset in pixel values and is calculated as:

$$MAE(y, \hat{y}) = |y - \hat{y}|. \tag{1}$$

The normalized root mean-squared error (NRMSE) compares the pixel values of the reconstruction $\hat{y}$ to the ground truth image y. NRMSE normalizes the root mean-squared error to account for the scale of the data, making it an scalar quantity that is easier to interpret.

$$NRMSE(y, \hat{y}) = \frac{\sqrt{MSE(y, \hat{y})}}{\|y\|_2} \tag{2}$$

Lower NRMSE values indicate a higher correspondence between the ground truth and reconstruction.

The peak signal-to-noise ratio (PSNR) quantifies the quality of reconstructed images using a logarithmic measure of the peak error (mean squared error, MSE) between y and $\hat{y}$. The PSNR value is expressed in decibels (dB), which logarithmically measures the ratio between the maximum possible pixel value L of the images (here $L = 255$) and the MSE:

$$PSNR(y, \hat{y}) = 10log_{10}\frac{L^2}{MSE} \tag{3}$$

Higher PSNR values indicate better image quality, suggesting that the reconstructed image is closer to the original image.

The structural similarity index measure (SSIM) [11] was designed to improve PSNR or MAE by also incorporating differences in luminance $l(y, \hat{y})$, contrast $c(y, \hat{y})$, and structural information $s(y, \hat{y})$. The SSIM is defined as:

$$SSIM(y, \hat{y}) = [l(y, \hat{y})]^{\alpha} \cdot [c(y, \hat{y})]^{\beta} \cdot [s(y, \hat{y})]^{\gamma} \tag{4}$$

where α, β, and γ define the relative importance of the three components. Here, we set all to 1 to equally weight each component. The SSIM ranges from 0 (structural dissimilarity) to 1 (perfect structural similarity). The multiscale SSIM (MS-SSIM) additionally evaluates the structural similarity across various scales to capture both fine details and coarse structures [12]. To this aim, the images are iteratively smoothed using a Gaussian low-pass filter and downsampled by a factor of 2. The SSIM is computed at each scale and the final MS-SSIM score is a weighted product of the SSIM scores of each scale. The weights emphasize different scales based on their importance to human perception. The MS-SSIM ranges from 0 (structural dissimilarity) to 1 (perfect structural similarity).

The *resolution*, as defined by [1], assesses the resolution of individual images based on decorrelation analysis. The core idea is to examine how the frequency components of the image decorrelate as the distance between them increases, in order to determine the point where significant loss of detail occurs, thereby defining the resolution of the image. High-resolution images have more details and, therefore, higher decorrelation between neighboring pixels. To compute the resolution first standard edge apodization is applied to the image to remove high-frequency artifacts. Then the image is Fourier transformed as $I(\mathbf{k})$, where $\mathbf{k} = [k_x, k_y]$ represent the coordinates in Fourier space. Additionally, the Fourier transform is normalized as $I_n(\mathbf{k}) = \frac{I(\mathbf{k})}{|I(\mathbf{k})|}$. Next, the cross-correlation between $I(\mathbf{k})$ and $I_n(\mathbf{k})$ is computed using the Pearson correlation and rescaled to a value between 0 and 1. The calculation is repeated but $I_n(\mathbf{k})$ is additionally filtered with a binary circular mask of radius $M(\mathbf{k}; r)$ with $r \in [0, 1]$. We can then compute the correlation coefficient as:

$$d(r) = \frac{\iint Re\{I(\mathbf{k}) I_n(\mathbf{k}) M(\mathbf{k}; r)\} dk_x dk_y}{\sqrt{\iint |I(\mathbf{k})|^2 dk_x dk_y \iint |I_n(\mathbf{k}) M(\mathbf{k}; r)|^2 dk_x dk_y}} \tag{5}$$

For differently high-pass filtered images (from weak to very strong filtering) $d(r)$ is computed and the peak position r_i and amplitude A_i are extracted. The resolution is then defined as the maximum peak across N_g high-pass filters as:

$$R = \frac{2 \times pixelsize}{max[r_0, \ldots, r_{N_g}]} \tag{6}$$

Lower values indicate a better resolution, as more fine-grained features are visible.

As the resolution is measured on each image individually, we propose a method for denoising tasks that computes the performance, respectively to the high-resolution data. Specifically, we compute:

$$\bar{R} = \frac{R_{\hat{y}}}{R_y} \tag{7}$$

where R_y and $R_{\hat{y}}$ refer to the resolution of the high-intensity image y and predicted image $\hat{y}$, respectively. Values close to 1 indicate similar resolution between the high-intensity image and the prediction, i.e. $R_y \approx R_{\hat{y}}$. Values above (resp. below) 1 indicate that the prediction exhibits worse (resp. better) resolution than the ground-truth high-intensity image.

The learned perceptual image patch similarity (LPIPS) [14] assesses the perceptual similarity between images. In contrast to PSNR and SSIM, LPIPS compares feature representations extracted from a pre-trained deep neural network (here AlexNet) to assess perceptual similarity, which often aligns more closely with human visual perception. The LPIPS value ranges from 0 (high perceptual similarity) to 1 (low perceptual similarity).

S3. Benchmark models

Here, we describe the specific setup and training conditions for each benchmark model.

- Noise2Void [5] - We use the TensorFlow implementation from the authors. Epochs: 100, batch size: 32, initial learning rate: 2e−4. All other parameters use the default. We use the best-trained state identified by default by Noise2Void.

- pix2pix [8] - We use the implementation from ZeroCostDL4Mic [9]. Epochs: 5, batch size: 1, initial learning rate: 2e−4.

- UNet-RCAN [2] - Default settings. Max epochs: 200, initial learning rate: 1e−4, batch size: 1. We use the best-trained state identified by UNet-RCAN.

- CARE [13] - We use the implementation from ZeroCostDL4Mic. Epochs: 1000, batch size: 8, initial learning rate: $4e{-}4$. We used the best-trained state identified by default by CARE.

S4. Versions

To compute the mean absolute error (MSE) and Pearson correlation, we use NumPy v1.24.4 [3]. The peak signal-to-noise ratio (PSNR) is computed using the scikit-image library v0.19.3 [10]. The multi-scale structural similarity index measure (MS-SSIM) and learned perceptual image patch similarity (LPIPS) are computed using Torchmetrics v1.3.1. The resolution is computed using the plugin ImageDecorrelationAnalysis [1] for ImageJ [7].

S5. Averaging across many reconstructions

To improve the performance of the DDPM and remove any noise that was not removed by the denoising process, we employ an averaging strategy. Specifically, we generate several images using the same conditioning input but different inference runs. We consistently observe an increase in performance across several metrics when averaging, except for LPIPS (see Fig. S1), and in some cases resolution (see DDPM vs. DDPM-avg for microtubule and synapse in Fig. S5). This might be explained by the smoothing effect of averaging which removes fine-grained structures. Note that this fine-grained structure is not always desirable to keep in the image and might also indicate noise. Moreover, we observe the performance saturating with approximately 10 averaged samples.

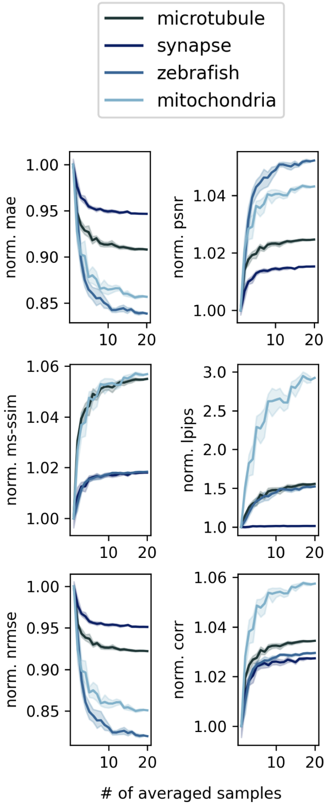

Figure S1. **Averaging across samples improves the performance for most metrics for the DDPM**. We repeatedly predict a denoised image using the same low-intensity conditioning input but different initial noise. We compute the mean image across different numbers of reconstructions. Average performance is shown in bold, and the translucent ban indicates the standard deviation.

S5.1. Uncertainty maps

We benefit from the above-described repeated sampling strategy to enhance the interpretability of the model. In particular, repeated sampling is valuable as it captures the variability of the model, thus reflecting its uncertainty in restoring certain areas of the image. After the model performs inference multiple times with the same conditioning input but different inference runs, we approximate the uncertainty based on the pixel-wise standard deviation (Eq. 8), and on the pixel-wise entropy (Eq. 9) across the different model outputs. In principle, it is also possible to compute uncertainty in a more abstract-fashion using the latent representations of the predicted image, i.e. in the $\mathcal{H}$-space of diffusion models, which we leave for future work.

Uncertainty maps provide us with a tool to verify that the model has learnt to restore regions in the image acc. For instance, one would expect complex and inherently ambiguous areas such as edges, to be predicted with a high uncertainty, otherwise suggesting over-fitting. Likewise, simple and smooth regions are expected to be predicted with low uncertainty, otherwise a sign of potential under-fitting. Additionally, if one were to collect additional data to refine the model, uncertainty maps can pinpoint the sub-structures that the current model struggles with, thus enabling a more informed data collection.

Additional to elucidating potential areas of improvement in the model, uncertainty maps can also be useful during the post-processing of the data, by informing about regions that could require further visual inspection or manual processing.

Let S be an uncertainty map for a set of images, each of size 256×256. Let s_{jk} be the map's pixel value located at (j,k). Thus, $S = (s_{jk})_{1\leq j\leq 256, 1\leq k\leq 256}$.

Given N repeated predictions $\{\hat{y}^1, ..., \hat{y}^N\}$ from the same noisy image, we compute the standard deviation-based uncertainty map as:

$$s_{jk} = \sqrt{\frac{\sum_{i=1}^{N}\left(\hat{y}^i_{jk} - \bar{\hat{y}}_{jk}\right)^2}{255^2 N}}, \tag{8}$$

where N = 15 is the number of times we repeat the sampling, $\bar{\hat{y}}_{jk}$ is the average of the multiple predicted samples at the (j,k)-th pixel, and 255^2 is a normalization factor to constrain values between 0 and 1.

Another way to compute uncertainty is based on the entropy of the pixel values across predicted samples, and is defined as follows:

$$s_{jk} = -\sum_{m=1}^{M} p_m \log p_m, \tag{9}$$

where M is the number of unique pixel values at location (j, k) among the single image predictions, and p_m is the probability of the m-th unique pixel value at location (j, k). We illustrate several examples of the two aforementioned types of uncertainty maps in Fig. S2.

When computing uncertainty as the pixel-wise standard deviation, we find that many high uncertainty regions correspond to the brighter areas of the low-resolution images. This might be due to small variations in intensity being amplified when calculating their difference. Another factor that could explain the higher uncertainty in bright regions is the complex structure underlying these areas, making their reconstruction more challenging for the model. Additionally, the model could be over-relying on these bright features to reconstruct the multiple samples, which would indicate a bias in how the model handles intensity features. Moreover, the model shows the highest uncertainty for the synapse dataset (see Fig. S2C), whereas the mitochondria dataset has the lowest uncertainty values (see Fig. S2B. In particular, for mitochondria, the model is most uncertain in predicting the membrane, an area which is inherently ambiguous in the noisy data (see Fig. S2B).

In contrast, uncertainty regions for the entropy-based formulation go beyond bright areas, and also include very noisy background regions. Combined with the previous observations, this can be interpreted as the predicted pixel intensities being uniformly distributed in a narrow range of values, which is a positive feature given the absence of complex structures on those regions, and namely the case for the background in the microtubules and the mitochondria datasets (see Fig. S2A, B). Furthermore, on the zebrafish images, we observe high uncertainty also in regions with visibly fine-grained details in the high-resolution image, that are ambiguous in the low-resolution image due to overlaid noise (see Fig. S2D). Thus, the model has not learnt to restore such small structures from noisy images.

In both uncertainty formulations, smooth regions in the noisy images are characterized by high-confidence values in the uncertainty maps, which reflects the model's ability to reliably predict non-complex regions.

S6. Results on additional metrics

Additionally to the results reported in the main text, we include additional metrics here (Fig. S3). Specifically, we report the performance of all models on the NRMSE and Pearson correlation for the internal (Table S3) and external (Table S4) datasets.

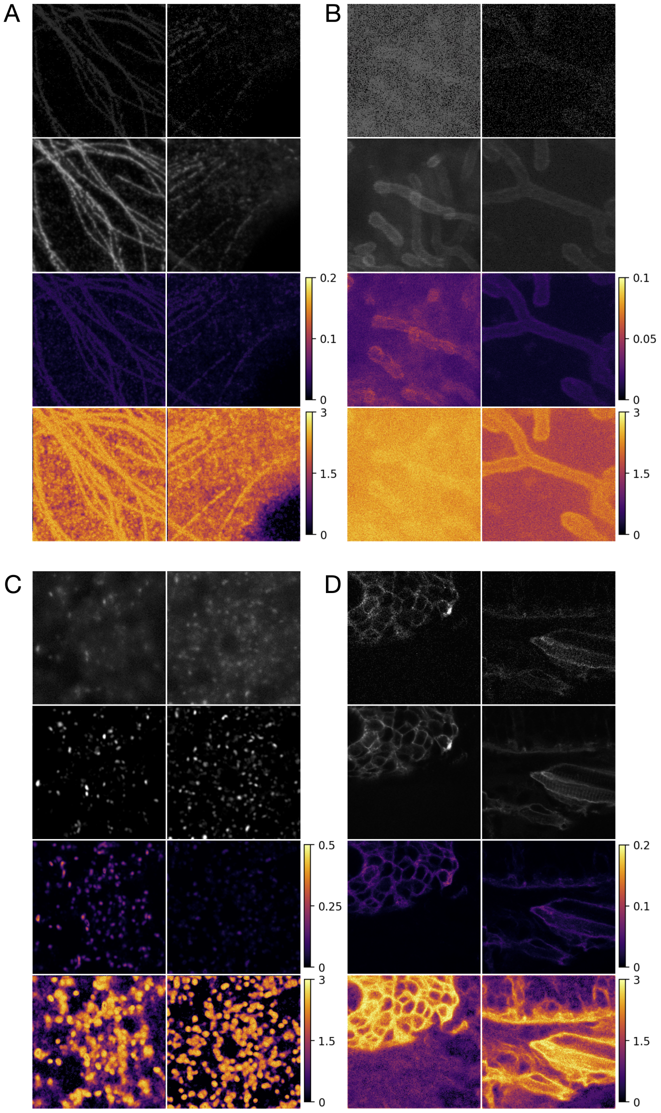

Figure S2. **Uncertainty maps based on repeated sampling strategy with DDPM.** For each dataset (**A**: microtubules, **B**: mitochondria, **C**: synapse, **D**: zebrafish), we show a subfigure with two low- (first row) and high- (second row) resolution images, and the resulting uncertainty maps, based on pixel-wise standard deviation (second row) or on entropy (third row). Note that for better visibility, the standard deviation-based uncertainty range is different for every dataset. Likewise, the pixel range was adjusted for the noisy images of mitochondria and microtubules.

S6.1. Reconstruction resolution

Additionally to the above-reported performance metrics, we also compute the resolution as proposed by Descloux et al. [1], as well as the resolution of the reconstruction scaled by that of the ground truth (resolution ratio; see Table S5). The resolution indicates the scale of the smallest fine-grained structure visible in the image. We observe that

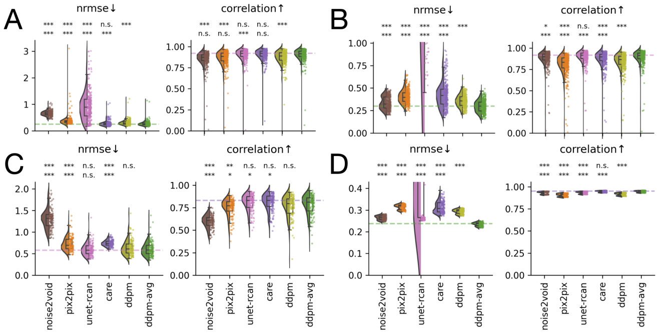

Figure S3. **Conditioned DDPMs outperform several previous methods in denoising STED and confocal images.** Performance comparison on additional metrics between our method and several previously proposed benchmark models for the microtubule (**A**), mitochondria (**B**), synapse (**C**), and zebrafish (**D**) datasets. We indicate the median of the best-performing model for each metric as a dashed line in the respective color. Mood's median test was used to compute statistical significance, ***: $p < .001$, **: $p < .01$, *: $p < .05$, otherwise not significant. In the upper (resp. lower) row, significance is indicated for the DDPM-avg (resp. DDPM).

	Microtubule		Mitochondria	
Model	NRMSE	Corr.	NRMSE	Corr.
Raw	0.99	0.46	0.97	0.40
Noise2Void	0.65	0.87	0.32	0.90
Pix2pix	0.35	0.88	0.40	0.83
UNet-RCAN	0.90	**0.92**	3.76	**0.92**
CARE	0.26	**0.92**	0.42	0.90
DDPM	0.29	0.89	0.36	0.87
DDPM-avg	**0.25**	**0.92**	**0.30**	**0.92**

Table S3. **Benchmarking the conditioned DDPM with additional metrics.** We report the median value of additional performance metrics, NRMSE (the lower the better) and Pearson correlation (the higher the better), across our two novel datasets.

	Synapse		Zebrafish	
Model	NRMSE	Corr.	NRMSE	Corr.
Raw	1.33	0.60	0.70	0.74
Noise2Void	1.32	0.61	0.27	0.94
Pix2pix	0.69	0.77	0.32	0.91
UNet-RCAN	**0.58**	**0.83**	0.55	0.94
CARE	0.74	**0.83**	0.31	**0.95**
DDPM	0.61	0.80	0.30	0.92
DDPM-avg	**0.58**	0.81	**0.24**	**0.95**

Table S4. **Benchmarking the conditioned DDPM with additional metrics.** Perfomance evaluation with NRMSE (the lower the better) and Pearson correlation (the higher the better) across the two external datasets.

pix2pix performs best for the fixed-cell microtubules and zebrafish datasets, Noise2Void on the synapse dataset, and UNet-RCAN on the live-cell mitochondria dataset. In particular, the resolution for the low-resolution images (raw) is lower than the high-resolution images (GT), suggesting the presence of artifacts, which is misleading for the evaluation of this metric for the synapse dataset. Note that all other evaluation metrics rate these methods poorly on the respective datasets. However, these metrics mostly rely on some form of pixel-wise error, whereas the resolution is based on cross-correlations within the image in the frequency domain. However, we observe that the resolution often picks up high-frequency noise in the data which wrongly improves the results.

	Microtubule	Mitochondria	Synapse	Zebrafish
Model	r / r ratio	r / r ratio	r / r ratio	r / r ratio
Raw	128.60 / 1.3	3563.64 / 11.91	143.14 / 0.49	5297.4 / 6.82
GT	98.80 / 1.00	299.24 / 1.00	293.33 / 1.00	776.70 / 1.00
Noise2Void	107.85 / 1.09	111.36 / 0.37	**147.04** / 0.50	1141.8 / 1.47
Pix2pix	**88.45** / 0.90	149.62 / 0.50	230.58 / 0.79	**730.05** / 0.94
UNet-RCAN	118.35 / 1.20	**76.72** / 0.27	385.88 / 1.32	1031.70 / 1.33
CARE	119.75 / 1.21	137.54 / 0.46	363.20 / 1.24	772.35 / 0.99
DDPM	97.6 / 0.99	177.38 / 0.59	330.18 / 1.13	831.60 / 1.07
DDPM-avg	115.28 / 1.17	110.74 / 0.37	363.20 / 1.24	777.75 / 1.00

Table S5. **Resolution across models and datasets.** We report the median of image resolution in nm, and the resolution ratio with respect to ground-truth (GT) resolution.

S7. Model architecture

S7.1. Timestep embedding

As in [4], we replace ADM's original timestep embedding layer, and instead embed the noise level information as Fourier features:

$$MPFourier(a) = \begin{bmatrix} \sqrt{2}cos(2\pi(f_1 a + \varphi_1)) \\ \sqrt{2}cos(2\pi(f_2 a + \varphi_2)) \\ \vdots \\ \sqrt{2}cos(2\pi(f_N a + \varphi_N)) \end{bmatrix}, \qquad (10)$$

where $f_i \sim \mathcal{N}(0,1)$, $\varphi \sim \mathcal{U}(0,1)$, and $a = \bar{\alpha}_t$ is a scalar defined as a function of the noise level t and the variance schedule. In the feature vector, $\sqrt{2}$ is the scaling factor that enables magnitude preservation, followed by a linear transformation (as shown in Fig. S4A) with learnable parameters, a magnitude-preserving sum operator, and a magnitude-preserving SiLU non-linearity.

S8. Statistical significance of model performances

To compare the performance across models we computed the p-values using Mood's median test. We report

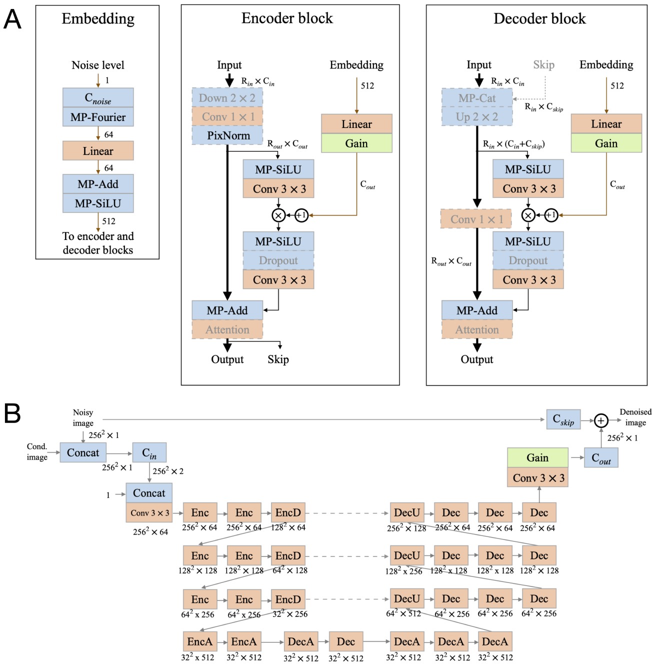

Figure S4. **U-Net architecture.** Adapted from Karras et al. [4]. **A**) We depict the three main parts of the U-Net model: an auxiliary embedding network that conditions the U-Net according to the noise level, encoder blocks that gradually decrease the resolution of the image, and decoding blocks that gradually increase it. **B**) The network receives as input the noisy image concatenated to the conditioning image (low-resolution image in our case). This is then processed by the encoder and decoder blocks following the main path (solid arrows), that additionaly communicate between them via skip connections (dashed arrows). *EncD* and *EncA* are encoder blocks that include downsampling and self-attention, respectively. This is analogous to decoder blocks *DecD* and *DecA*. c_{in}, c_{out}, c_{skip} are constants that depend on the noise level. MP stands for Magnitude-Preserving. Layers are color-coded as follows: green - parameters are learned, clay - parameters are learned with *forced weight normalization*, blue - function is fixed, dashed contour - not always present.

them here below (see Tables S6, S7, S8, S9) for each dataset and all metrics.

Model 1	Model 2	Metric	p
DDPM	Noise2Void	MAE	5.37e-41
DDPM	pix2pix	MAE	5.97e-01
DDPM	UNet-RCAN	MAE	1.91e-30
DDPM	CARE	MAE	1.11e-16
DDPM-avg	Noise2Void	MAE	2.31e-54
DDPM-avg	pix2pix	MAE	4.82e-16
DDPM-avg	UNet-RCAN	MAE	3.56e-53
DDPM-avg	CARE	MAE	3.78e-01
DDPM-avg	DDPM	MAE	5.37e-18
DDPM	Noise2Void	PSNR	5.70e-40
DDPM	pix2pix	PSNR	1.12e-01
DDPM	UNet-RCAN	PSNR	3.09e-45
DDPM	CARE	PSNR	4.54e-20
DDPM-avg	Noise2Void	PSNR	3.01e-59
DDPM-avg	pix2pix	PSNR	2.48e-17
DDPM-avg	UNet-RCAN	PSNR	4.97e-71
DDPM-avg	CARE	PSNR	3.78e-01
DDPM-avg	DDPM	PSNR	4.54e-20
DDPM	Noise2Void	MS-SSIM	6.61e-10
DDPM	pix2pix	MS-SSIM	2.02e-11
DDPM	UNet-RCAN	MS-SSIM	3.42e-02
DDPM	CARE	MS-SSIM	5.33e-36
DDPM-avg	Noise2Void	MS-SSIM	5.24e-58
DDPM-avg	pix2pix	MS-SSIM	2.11e-04
DDPM-avg	UNet-RCAN	MS-SSIM	4.82e-16
DDPM-avg	CARE	MS-SSIM	5.97e-01
DDPM-avg	DDPM	MS-SSIM	4.28e-34
DDPM	Noise2Void	LPIPS	3.71e-44
DDPM	pix2pix	LPIPS	1.35e-02
DDPM	UNet-RCAN	LPIPS	5.37e-41
DDPM	CARE	LPIPS	3.46e-26
DDPM-avg	Noise2Void	LPIPS	2.11e-04
DDPM-avg	pix2pix	LPIPS	2.13e-10
DDPM-avg	UNet-RCAN	LPIPS	4.95e-05
DDPM-avg	CARE	LPIPS	7.24e-01
DDPM-avg	DDPM	LPIPS	5.18e-27

Table S6. **P-values reported for the microtubule dataset.**

Model 1	Model 2	Metric	p
DDPM	Noise2Void	MAE	7.63e-03
DDPM	pix2pix	MAE	7.45e-05
DDPM	UNet-RCAN	MAE	3.04e-134
DDPM	CARE	MAE	1.83e-05
DDPM-avg	Noise2Void	MAE	4.66e-03
DDPM-avg	pix2pix	MAE	2.09e-17
DDPM-avg	UNet-RCAN	MAE	1.63e-132
DDPM-avg	CARE	MAE	1.23e-18
DDPM-avg	DDPM	MAE	3.52e-07
DDPM	Noise2Void	PSNR	1.91e-02
DDPM	pix2pix	PSNR	1.62e-03
DDPM	UNet-RCAN	PSNR	3.04e-134
DDPM	CARE	PSNR	1.84e-06
DDPM-avg	Noise2Void	PSNR	1.62e-03
DDPM-avg	pix2pix	PSNR	5.13e-18
DDPM-avg	UNet-RCAN	PSNR	3.04e-134
DDPM-avg	CARE	PSNR	6.51e-20
DDPM-avg	DDPM	PSNR	4.81e-10
DDPM	Noise2Void	MS-SSIM	8.08e-01
DDPM	pix2pix	MS-SSIM	1.62e-03
DDPM	UNet-RCAN	MS-SSIM	1.64e-28
DDPM	CARE	MS-SSIM	1.25e-01
DDPM-avg	Noise2Void	MS-SSIM	3.74e-05
DDPM-avg	pix2pix	MS-SSIM	5.13e-18
DDPM-avg	UNet-RCAN	MS-SSIM	1.42e-50
DDPM-avg	CARE	MS-SSIM	9.18e-04
DDPM-avg	DDPM	MS-SSIM	8.16e-07
DDPM	Noise2Void	LPIPS	3.04e-134
DDPM	pix2pix	LPIPS	7.63e-03
DDPM	UNet-RCAN	LPIPS	3.04e-134
DDPM	CARE	LPIPS	6.49e-37
DDPM-avg	Noise2Void	LPIPS	1.48e-07
DDPM-avg	pix2pix	LPIPS	2.13e-106
DDPM-avg	UNet-RCAN	LPIPS	7.49e-116
DDPM-avg	CARE	LPIPS	1.82e-117
DDPM-avg	DDPM	LPIPS	1.63e-132

Table S7. **P-values reported for the mitochondria dataset.**

Model 1	Model 2	Metric	p
DDPM	Noise2Void	MAE	1.00e-44
DDPM	pix2pix	MAE	8.35e-04
DDPM	UNet-RCAN	MAE	2.29e-01
DDPM	CARE	MAE	1.22e-23
DDPM-avg	Noise2Void	MAE	1.00e-44
DDPM-avg	pix2pix	MAE	3.43e-05
DDPM-avg	UNet-RCAN	MAE	2.31e-02
DDPM-avg	CARE	MAE	1.22e-23
DDPM-avg	DDPM	MAE	2.29e-01
DDPM	Noise2Void	PSNR	3.04e-31
DDPM	pix2pix	PSNR	5.01e-03
DDPM	UNet-RCAN	PSNR	5.04e-01
DDPM	CARE	PSNR	4.28e-08
DDPM-avg	Noise2Void	PSNR	1.29e-32
DDPM-avg	pix2pix	PSNR	3.09e-04
DDPM-avg	UNet-RCAN	PSNR	8.94e-01
DDPM-avg	CARE	PSNR	9.13e-09
DDPM-avg	DDPM	PSNR	5.04e-01
DDPM	Noise2Void	MS-SSIM	2.65e-27
DDPM	pix2pix	MS-SSIM	2.31e-02
DDPM	UNet-RCAN	MS-SSIM	5.04e-01
DDPM	CARE	MS-SSIM	9.42e-12
DDPM-avg	Noise2Void	MS-SSIM	1.38e-28
DDPM-avg	pix2pix	MS-SSIM	3.09e-04
DDPM-avg	UNet-RCAN	MS-SSIM	6.88e-01
DDPM-avg	CARE	MS-SSIM	1.99e-13
DDPM-avg	DDPM	MS-SSIM	5.04e-01
DDPM	Noise2Void	LPIPS	2.24e-46
DDPM	pix2pix	LPIPS	2.29e-01
DDPM	UNet-RCAN	LPIPS	8.24e-02
DDPM	CARE	LPIPS	3.37e-10
DDPM-avg	Noise2Void	LPIPS	2.24e-46
DDPM-avg	pix2pix	LPIPS	5.04e-01
DDPM-avg	UNet-RCAN	LPIPS	1.42e-01
DDPM-avg	CARE	LPIPS	1.82e-09
DDPM-avg	DDPM	LPIPS	8.94e-01

Table S8. **P-values reported for the synapse dataset.**

Model 1	Model 2	Metric	p
DDPM	Noise2Void	MAE	2.14e-10
DDPM	pix2pix	MAE	2.51e-36
DDPM	UNet-RCAN	MAE	2.43e-66
DDPM	CARE	MAE	9.36e-10
DDPM-avg	Noise2Void	MAE	2.43e-66
DDPM-avg	pix2pix	MAE	2.43e-66
DDPM-avg	UNet-RCAN	MAE	2.43e-66
DDPM-avg	CARE	MAE	2.43e-66
DDPM-avg	DDPM	MAE	2.43e-66
DDPM	Noise2Void	PSNR	1.53e-08
DDPM	pix2pix	PSNR	1.53e-08
DDPM	UNet-RCAN	PSNR	1.53e-08
DDPM	CARE	PSNR	5.64e-01
DDPM-avg	Noise2Void	PSNR	1.53e-08
DDPM-avg	pix2pix	PSNR	6.35e-63
DDPM-avg	UNet-RCAN	PSNR	2.43e-66
DDPM-avg	CARE	PSNR	2.43e-66
DDPM-avg	DDPM	PSNR	1.02e-14
DDPM	Noise2Void	MS-SSIM	5.73e-08
DDPM	pix2pix	MS-SSIM	6.69e-06
DDPM	UNet-RCAN	MS-SSIM	1.53e-08
DDPM	CARE	MS-SSIM	5.73e-08
DDPM-avg	Noise2Void	MS-SSIM	1.53e-08
DDPM-avg	pix2pix	MS-SSIM	1.53e-08
DDPM-avg	UNet-RCAN	MS-SSIM	1.53e-08
DDPM-avg	CARE	MS-SSIM	5.73e-08
DDPM-avg	DDPM	MS-SSIM	1.53e-08
DDPM	Noise2Void	LPIPS	2.14e-10
DDPM	pix2pix	LPIPS	3.44e-04
DDPM	UNet-RCAN	LPIPS	2.43e-66
DDPM	CARE	LPIPS	8.96e-25
DDPM-avg	Noise2Void	LPIPS	1.53e-08
DDPM-avg	pix2pix	LPIPS	8.51e-21
DDPM-avg	UNet-RCAN	LPIPS	1.53e-08
DDPM-avg	CARE	LPIPS	8.12e-04
DDPM-avg	DDPM	LPIPS	8.96e-25

Table S9. **P-values reported for the zebrafish dataset.**

References

[1] A. Descloux, K. S. Grußmayer, and A. Radenovic. Parameter-free image resolution estimation based on decorrelation analysis. *Nature Methods*, 16(9):918–924, Sept. 2019. Publisher: Nature Publishing Group. 2, 3, 4

[2] Vahid Ebrahimi, Till Stephan, Jiah Kim, Pablo Carravilla, Christian Eggeling, Stefan Jakobs, and Kyu Young Han. Deep learning enables fast, gentle STED microscopy. *Communications Biology*, 6(1):1–8, June 2023. Publisher: Nature Publishing Group. 2

[3] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. Publisher: Nature Publishing Group. 3

[4] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and Improving the Training Dynamics of Diffusion Models, Mar. 2024. arXiv:2312.02696 [cs, stat]. 5, 6

[5] Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2Void - Learning Denoising from Single Noisy Images, Apr. 2019. arXiv:1811.10980 [cs]. 2

[6] Matthew Michael McCormick, Xiaoxiao Liu, Luis Ibanez, Julien Jomier, and Charles Marion. ITK: enabling reproducible research and open science. *Frontiers in Neuroinformatics*, 8, Feb. 2014. Publisher: Frontiers. 1

[7] Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, Jean-Yves Tinevez, Daniel James White, Volker Hartenstein, Kevin Eliceiri, Pavel Tomancak, and Albert Cardona. Fiji: an open-source platform for biological-image analysis. *Nature Methods*, 9(7):676–682, July 2012. Publisher: Nature Publishing Group. 3

[8] Jingzhang Sun, Yu Du, Chien-Ying Li, Tung-Hsin Wu, Bang-Hung Yang, and Greta S. P. Mok. Pix2Pix generative adversarial network for low dose myocardial perfusion SPECT denoising. *Quantitative Imaging in Medicine and Surgery*, 12(7):3539555–3533555, July 2022. Publisher: AME Publishing Company. 2

[9] Lucas von Chamier, Romain F. Laine, Johanna Jukkala, Christoph Spahn, Daniel Krentzel, Elias Nehme, Martina Lerche, Sara Hernández-Pérez, Pieta K. Mattila, Eleni Karinou, Séamus Holden, Ahmet Can Solak, Alexander Krull, Tim-Oliver Buchholz, Martin L. Jones, Loïc A. Royer, Christophe Leterrier, Yoav Shechtman, Florian Jug, Mike Heilemann, Guillaume Jacquemet, and Ricardo Henriques. Democratising deep learning for microscopy with ZeroCostDL4Mic. *Nature Communications*, 12(1):2276, Apr. 2021. Publisher: Nature Publishing Group. 2

[10] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in Python. *PeerJ*, 2:e453, June 2014. Publisher: PeerJ Inc. 3

[11] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, Apr. 2004. Conference Name: IEEE Transactions on Image Processing. 2

[12] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402 Vol.2, Nov. 2003. 2

[13] Martin Weigert, Uwe Schmidt, Tobias Boothe, Andreas Müller, Alexandr Dibrov, Akanksha Jain, Benjamin Wilhelm, Deborah Schmidt, Coleman Broaddus, Siân Culley, Mauricio Rocha-Martins, Fabián Segovia-Miranda, Caren Norden, Ricardo Henriques, Marino Zerial, Michele Solimena, Jochen Rink, Pavel Tomancak, Loic Royer, Florian Jug, and Eugene W. Myers. Content-aware image restoration: pushing the limits of fluorescence microscopy. *Nature Methods*, 15(12):1090–1097, Dec. 2018. Number: 12 Publisher: Nature Publishing Group. 2

[14] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric, Apr. 2018. arXiv:1801.03924 [cs]. 2