

A fast blind zero-shot denoiser

Received: 17 February 2022

Accepted: 14 September 2022

Published online: 31 October 2022

 Check for updates

Jason Lequyer^{1,2}, Reuben Philip^{1,2}, Amit Sharma¹, Wen-Hsin Hsu¹ and Laurence Pelletier^{1,2}✉

Image noise is a common problem in light microscopy. This is particularly true in real-time live-cell imaging applications in which long-term cell viability necessitates low-light conditions. Modern denoisers are typically trained on a representative dataset, sometimes consisting of just unpaired noisy shots. However, when data are acquired in real time to track dynamic cellular processes, it is not always practical nor economical to generate these training sets. Recently, denoisers have emerged that allow us to denoise single images without a training set or knowledge about the underlying noise. But such methods are currently too slow to be integrated into imaging pipelines that require rapid, real-time hardware feedback. Here we present Noise2Fast, which can overcome these limitations. Noise2Fast uses a novel downsampling technique we refer to as ‘chequerboard downsampling’. This allows us to train on a discrete 4-image training set, while convergence can be monitored using the original noisy image. We show that Noise2Fast is faster than all similar methods with only a small drop in accuracy compared to the gold standard. We integrate Noise2Fast into real-time multi-modal imaging applications and demonstrate its broad applicability to diverse imaging and analysis pipelines.

Image noise is the random fluctuation of colour or intensity values that is inherent to image acquisition. It usually presents as a hazy shroud that obscures an otherwise clear visual signal. Image denoising methods try to fix this by removing noise after the fact, usually by exploiting the innate structure and pattern of the underlying signal and leveraging it against the apparent stochasticity of the noise¹. Denoising is particularly important in live-cell imaging applications, where a balance between the conflicting considerations of resolution, phototoxicity and throughput can force the acceptance of a considerable amount of noise to achieve experimental goals.

Many techniques focus on modelling noise by understanding its origin; for example, confocal microscopy is mainly subject to a combination of Gaussian- and Poisson-distributed noise². However, with deep learning, such explicit models are avoidable by instead training a neural network to learn to map noisy images to their clean counterparts, such as in DnCNN³, or even by training it to map noisy pairs of images to one another, such as in Noise2Noise⁴, both are essential components of the CARE toolbox⁵.

However, these methods cannot perform effectively on data that were not well represented in the training set, and hence the training set itself can become a source of bias and variation. Moreover, it is not always practical to acquire representative training data. One example of this is live automated microscopy where real-time image analysis is used to alter the behaviour of the imaging system. Such pipelines are usually confronted with single, noisy images and no additional context. While representative training data could be collected in a separate run, this is time consuming, resource intensive and requires expertise that is not always available. For these reasons, blind zero-shot denoisers have been developed.

Blind zero-shot denoisers train themselves on the very image they are trying to denoise, appealing to no other outside information or knowledge about the distribution and/or variance of the noise in the underlying image. Noise2Void⁶ is one of the earliest methods that can be tailored to achieve this. Noise2Void denoises images by using a masking procedure wherein the neural network learns to fill in pixel gaps in the noisy image. The failure of the network to learn the noise

¹Lunenfeld-Tanenbaum Research Institute, Toronto, Ontario, Canada. ²Department of Molecular Genetics, University of Toronto, Toronto, Ontario, Canada.

✉e-mail: pelletier@lunenfeld.ca

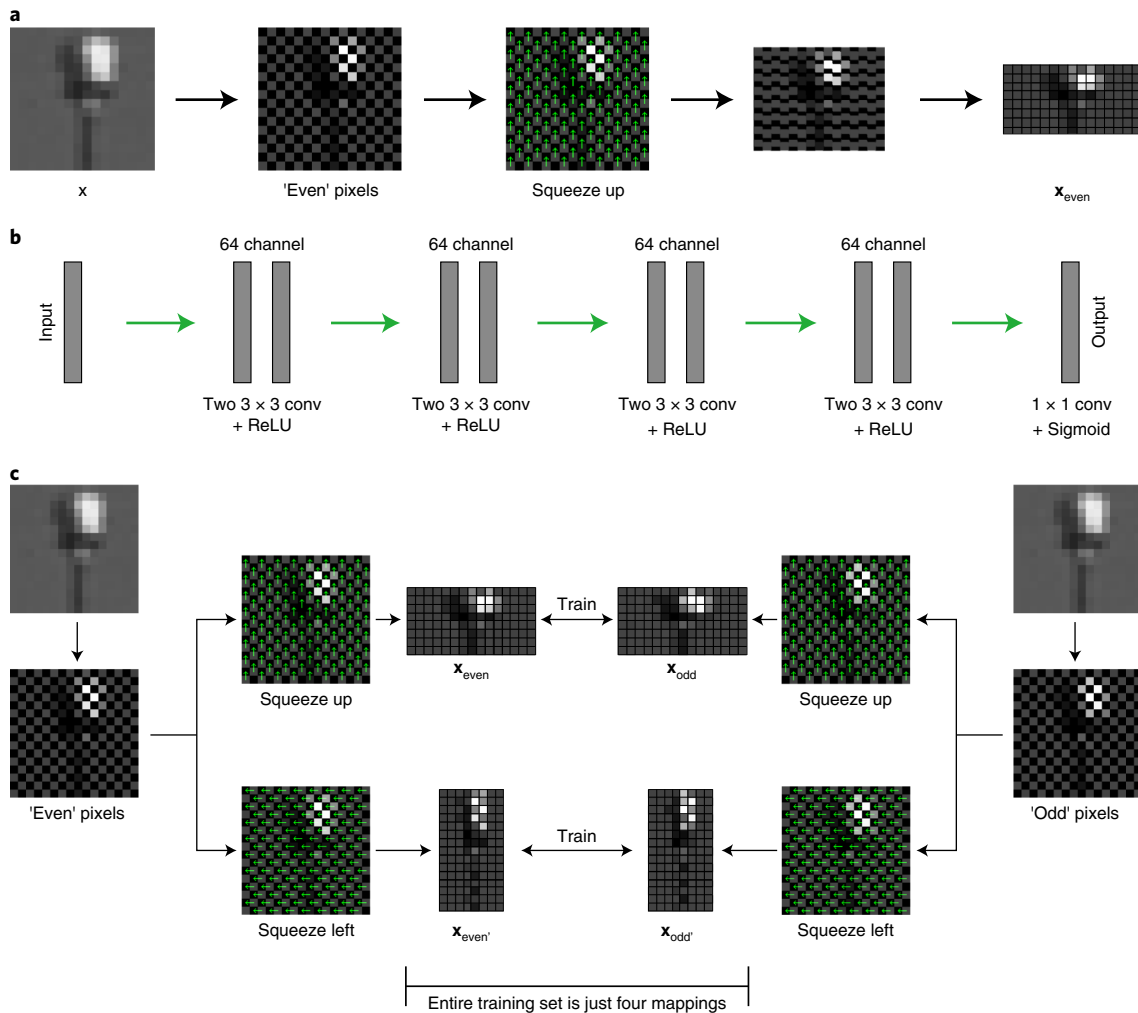


Fig. 1 | An overview of how Noise2Fast trains itself. **a**, Chequerboard downsampling illustrated. We take our initial image, remove one half of all pixels in a chequerboard pattern, and shift the remaining pixels to fill in the gaps left behind. **b**, Our simple neural network architecture. Inputs can be multi-channel,

however, for best results outputs are always single channel (for colour images we predict each channel separately). **c**, Overview of our training scheme. Our neural network learns mappings between pairs of chequerboard downsampled images, each generated from different group of pixels.

causes it to denoise the underlying image. Although it was trained on entire datasets of images with similar noise levels, Noise2Void can be adapted to denoise single noisy images by restricting the training set to just that image (and virtually every other ‘single-shot denoiser’ including Recorruped-to-Recorruped, Noise2Void and Noise2Self, can be adapted in a similar way). The basic idea of Noise2Void was improved and generalized in Noise2Self⁷ and further refined in Self2Self⁸ to achieve single image denoising results that are competitive with traditional fully trained methods. However, all viable blind zero-shot denoisers to date require a considerable amount of time to run (for example, Self2Self takes 4 hours to denoise a single 512×512 confocal image), making them impractical for use in real-time situations.

To alleviate this, we propose Noise2Fast. Our method is inspired by a recently published approach called Neighbor2Neighbor⁹ where the neural network learns a mapping between adjacent pixels. We tune our method to speed by using a discrete four-image training set obtained by an unusual form of downsampling we refer to as ‘chequerboard downsampling’ and train a small neural network on this discrete training set. Although such a method inevitably overfits, we can accurately validate using the original full-sized noisy image since our distorted downsamplings do not locally resemble it, giving us a method that has a natural cut-off time unlike blind-spot-based denoisers. Noise2Fast is

faster than all compared methods, and is more accurate than all tested methods except for Self2Self.

Consider a 2D image $\mathbf{x} \in \mathbb{R}^{m \times n}$ composed of both signal and noise $\mathbf{s}, \mathbf{n} \in \mathbb{R}^{m \times n}$. That is to say

$$\mathbf{x} = \mathbf{s} + \mathbf{n}. \tag{1}$$

Denoising is concerned with the inverse problem of inferring \mathbf{s} from \mathbf{x} (or equivalently inferring \mathbf{n} and then solving for \mathbf{s}). A neural network attempts to solve this problem by finding a function $f_\theta : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \times n}$ (parameterized by the network weights θ) such that

$$f_\theta(\mathbf{x}) \approx \mathbf{s}. \tag{2}$$

The most intuitive way to train such a network is by using pairs of noisy/clean images and having the network learn a mapping from one to the other. Noise2Noise trains the network to learn a mapping from different noisy shots of the same image, allowing for training in the absence of clean ground truth data. Specifically, given two noisy realizations of the same underlying signal $\mathbf{s} + \mathbf{n}_1$ and $\mathbf{s} + \mathbf{n}_2$, Noise2Noise attempts to learn the mapping

$$f_\theta(\mathbf{s} + \mathbf{n}_1) \rightarrow \mathbf{s} + \mathbf{n}_2. \tag{3}$$

Table 1 | Accuracy and speed of Noise2Fast

Dataset	σ	Noise2Self (single)		Noise2Void (single)		DIP3000		Neighbor2Neighbor (single)		Self2Self		Noise2Fast	
		PSNR/SSIM	Time per image	PSNR/SSIM	Time per image	PSNR/SSIM	Time per image	PSNR/SSIM	Time per image	PSNR/SSIM	Time per image	PSNR/SSIM	Time per image
Set12	15	30.69/8.71	2,161s	30.04/8.47	2,682s	28.51/8.04	70s	27.97/7.87	198s	32.17/8.89	9,484s	31.10/8.71	22s
	25	28.35/7.76	2,161s	28.41/7.80	2,682s	26.47/7.07	70s	26.23/6.87	198s	29.88/8.42	9,484s	29.05/8.22	18s
	35	26.59/7.27	2,161s	27.29/7.71	2,682s	24.25/5.96	71s	25.09/6.32	198s	28.24/7.99	9,484s	27.57/7.81	19s
	50	25.04/6.87	2,161s	25.70/7.20	2,682s	21.19/4.39	70s	23.43/5.32	198s	26.34/7.34	9,484s	25.82/7.23	21s
BSD68	25	27.50/7.73	1,619s	26.66/7.31	2,682s	25.74/6.85	69s	25.68/6.92	231s	28.70/8.03	7,962s	28.12/7.89	29s
	50	24.53/6.46	1,619s	24.50/6.05	2,682s	21.29/4.42	69s	23.59/5.45	231s	25.92/6.99	7,962s	25.23/6.70	26s
Confocal	-	36.45/9.31	4,015s	36.45/9.31	2,682s	35.16/9.05	102s	14.78/3.01	400s	36.99/9.38	18,016s	36.61/9.33	56s

PSNR and per-image time required to denoise on an RTX 5000 mobile GPU, for each dataset using each method.

However, if we assume mean-zero noise and choose a sensible loss function⁴, the network may fail to actually learn the noise \mathbf{n}_2 , and we will be left with

$$f_{\theta}(\mathbf{s} + \mathbf{n}_1) \approx \mathbf{s}, \tag{4}$$

denoising the image as a result. Although elegant, this method still requires pairs of noisy images to train on.

Recently, interest has grown in methods that can denoise single noisy images, without this added requirement. To fully understand these methods, we need to adopt a different perspective of how neural networks denoise images.

Here we take the view of Krull et al.⁶, based on the concept of receptive fields. The receptive field of a fully convolutional neural network (FCN) is the set of input pixels that were taken into consideration for a given output pixel prediction. For example, in our above scenario suppose $(i, j) \in \mathbb{N}_{\leq m} \times \mathbb{N}_{\leq n}$ are the co-ordinates of some pixel in the output image $f_{\theta}(\mathbf{x})$. Then the receptive field (RF) of that pixel is the set of indices $\text{RF}(i, j) \subseteq \mathbb{N}_{\leq m} \times \mathbb{N}_{\leq n}$ such that $f_{\theta}(\mathbf{x})_{(i, j)}$ depends only upon the value of $\mathbf{x}|_{\text{RF}(i, j)}$ (typically this will be a small square patch of the image \mathbf{x}). We can then view the neural network as a mapping from the input image along some receptive field to its corresponding output pixel, with the goal of finding θ such that

$$f_{\theta}(\mathbf{x}|_{\text{RF}(i, j)}) \approx \mathbf{s}(i, j), \tag{5}$$

for every $(i, j) \in \mathbb{N}_{\leq m} \times \mathbb{N}_{\leq n}$. The question though, is how to train these networks without any actual training data other than the noisy image itself. Blind-spot methods approach this by excluding the centre pixel from the receptive field (either by removing/replacing it^{6,7} or ignoring it altogether using partial convolutions⁸), and training the network to recover this centre pixel from its surroundings. More specifically, they train the network to learn the mapping,

$$f_{\theta}(\mathbf{x}|_{\text{RF}(i, j) \setminus (i, j)}) \rightarrow \mathbf{x}(i, j). \tag{6}$$

However, just as in Noise2Noise, the network fails to learn the noise, leaving us with

$$f_{\theta}(\mathbf{x}|_{\text{RF}(i, j) \setminus (i, j)}) \approx \mathbf{s}(i, j). \tag{7}$$

Excluding the centre pixel is crucial and ensures that the network does not just learn the identity. However, a side effect of this is that the neural network does not give proper weight to the pixel itself when computing the output, which is unfortunate, because

the pixel itself is always going to be the best individual predictor of its denoised value.

Our method takes a related, but slightly different approach. Instead of masking the input image, we explicitly divide the input image in two, by using a simple downsampling method that we refer to as chequerboard downsampling. This process is easier to visualize than explain (see Fig. 1a), however we take our input image \mathbf{x} and split it into two smaller images composed of the even pixels (where $i + j$ is even) and odd pixels (where $i + j$ is odd) respectively, and compress them into the two following $m \times \frac{1}{2}n$ images

$$\mathbf{x}_{\text{even}}(i, j) = \mathbf{x}(i, 2j + (i \bmod 2)), \tag{8}$$

$$\mathbf{x}_{\text{odd}}(i, j) = \mathbf{x}(i, 2j + (i \bmod 2) + 1). \tag{9}$$

We can call these the ‘up’ chequerboard downsamples, since they involve shifting everything up one pixel to close the image.

Now suppose we train our neural network (Fig. 1b) to learn the mapping

$$f_{\theta}(\mathbf{x}_{\text{even}}) \rightarrow \mathbf{x}_{\text{odd}}. \tag{10}$$

We can rewrite this as

$$f_{\theta}(\mathbf{s}_{\text{even}} + \mathbf{n}_{\text{even}}) \rightarrow \mathbf{s}_{\text{even}} + \mathbf{n}_{\text{odd}} + (\mathbf{s}_{\text{odd}} - \mathbf{s}_{\text{even}}). \tag{11}$$

Notice that this is analogous to Noise2Noise (equation (3)), except for the addition of the $(\mathbf{s}_{\text{odd}} - \mathbf{s}_{\text{even}})$ term. However, for every $(i, j) \in \mathbb{N}_{\leq m} \times \mathbb{N}_{\leq n}$ we have that $\mathbf{s}_{\text{odd}}(i, j)$ and $\mathbf{s}_{\text{even}}(i, j)$ are adjacent pixels in the original image signal, it is therefore reasonable to think this term would be very small in all but the most highly dynamic regions. Indeed, in our testing, we found that even if we cheat and subtract out the term using known ground truth values, there was no measurable gain in denoising performance. We therefore claim that for most natural images,

$$\mathbf{s}_{\text{even}} + \mathbf{n}_{\text{odd}} + (\mathbf{s}_{\text{odd}} - \mathbf{s}_{\text{even}}) \approx \mathbf{s}_{\text{even}} + \mathbf{n}_{\text{odd}}. \tag{12}$$

Then, analogous with Noise2Noise (equations (3) and (4)), training our network as outlined in equation (10) should, in effect, find weights θ such that

$$f_{\theta}(\mathbf{x}_{\text{even}}) \approx \mathbf{s}_{\text{even}}. \tag{13}$$

However, in our experiments we have witnessed a much stronger result than this. In particular, we observe that a network trained as in equation

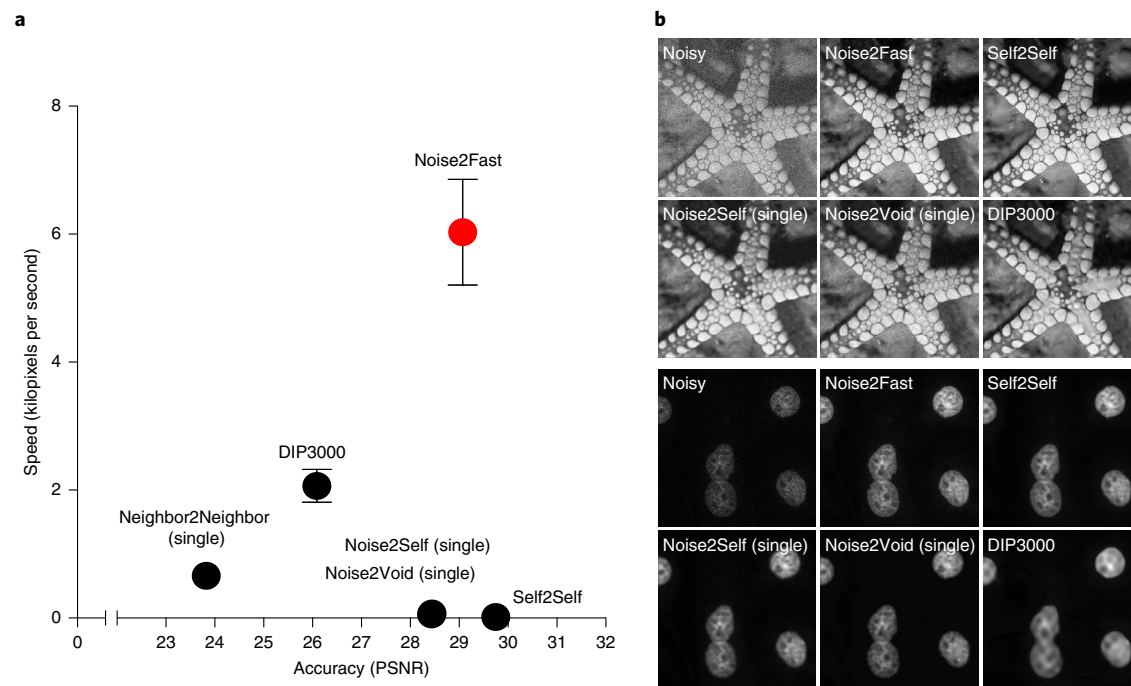


Fig. 2 | Benchmarking Noise2Fast. **a**, Graph of the speed (in kilopixels per second) of each method on each of the datasets. Error bars represent standard error with $n = 7$. **b**, Visual comparison of each method on starfish image from Set12 and on BPAE cells from Confocal dataset.

(10) will not just learn to denoise the downsampled image, but the entire image as a whole. That is

$$f_{\theta}(\mathbf{x}) \approx \mathbf{s}. \quad (14)$$

To explain this phenomenon, we return to the receptive field-based perspective of equation (6). In this case, our network is trained to learn the mapping

$$f_{\theta}(\mathbf{x}_{\text{even}}|_{\text{RF}(i,j)}) \rightarrow \mathbf{x}_{\text{odd}}(i,j). \quad (15)$$

It is known, and is often exploited by denoising algorithms, that single images contain significant internal redundancy in the form of recurrent patches¹⁰. It is also known, and is crucial to some super-resolution methods, that single images have a certain degree of self-similarity, and hence these patches also recur across scales^{11–14}. This across-scale patch recurrence implies similarity between the patches in the checkerboard downsampled images and the original full-sized image. We demonstrate this in Extended Data Fig. 1, by comparing the patch-wise similarity between an image and its checkerboard down-samplings. Hence, a neural network trained to learn as in equation (10) may be applicable to the overarching denoising task.

Our method uses this basic principle to generate a small training set of four-image pairs (Fig. 1c). This compact training set allows for rapid network convergence and hence quick single image denoising results that were previously unattainable with such a high degree of accuracy.

Contribution and significance

Our main contributions are as follows:

- **A novel denoising method that combines an unusual down-sampling method with Neighbor2Neighbor.** Our method uses checkerboard downsampling to generate a small four-image fixed dataset out of one single image. We then apply our network trained

on this smaller dataset to denoise the larger input image, which also serves as our validation set.

- **High accuracy and substantial speed gains over existing methods.** Our method is tailored specifically for speed; using a small four-image dataset ensures rapid convergence, and our validation strategy avoids overfitting. Our method is also quite accurate, in terms of PSNR (peak signal-to-noise ratio) and SSIM (Structural Similarity Index Measure), the only tested method more accurate than the one we propose here is Self2Self, which is an average of 200 times slower (Table 1 and Fig. 2).
- **The first blind single image denoiser that can viably be inserted into live automated microscopy pipelines.** We specifically demonstrate that our method can be seamlessly inserted into ‘smart’ imaging pipeline where the microscope monitors a large field of view containing dozens of cells, detects when a cell enters mitosis, and then zooms in on that cell for the duration of mitosis. Since the window to perform these analyses is only 5 minutes and denoising is just the first step in the pipeline, no other blind zero-shot denoiser could be feasibly placed here.

Related work

Methods that require a training set

The first attempt to apply convolutional neural networks (CNNs) to the task of denoising was in ref.¹⁵. This was heavily refined in refs.^{3,16} (DnCNN) to achieve performance that is still competitive today. Zhang et al. later released FFDNet¹⁷, a denoising CNN designed with speed in mind which, similar to our method, also uses downsampling, although in a different manner and to an entirely different end (see ref.¹⁸).

The main benefit of using trained methods, outside of their outstanding performance, is that they do not require assumptions about the type and structure of the noise, they can simply be trained on noisy/clean pairs of images. However, their reliance on noisy/clean image pairs can be considered a limitation in situations where we do not have access to ground truth images to train on.

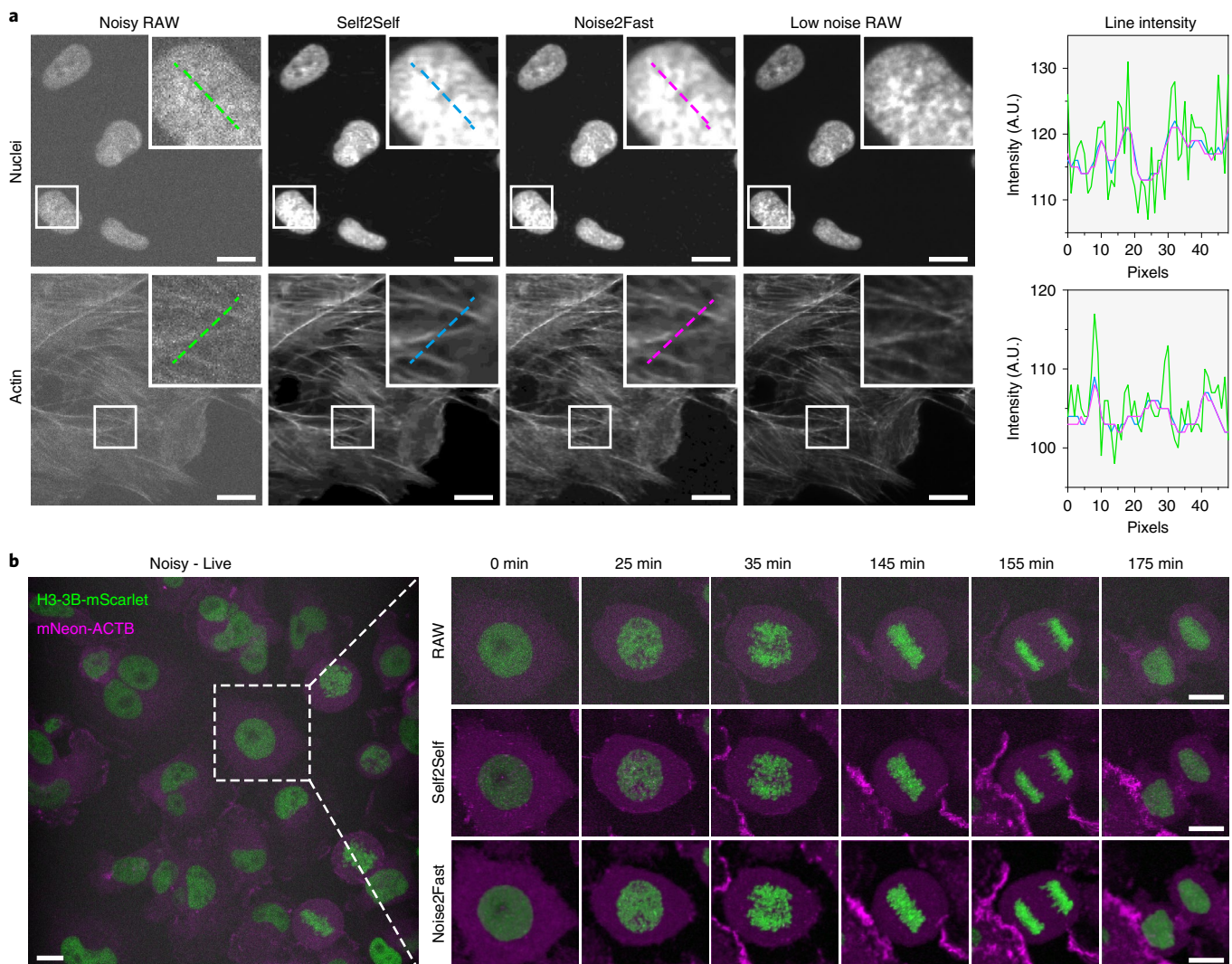


Fig. 3 | Performance of Noise2Fast on our own microscopy images. a, Comparison of Noise2Fast and Self2Self on epifluorescence images of actin and nuclei in RPE-1 cells with corresponding line intensity profiles. **b**, Comparison of live confocal imaging of endogenously tagged nuclei (H3-3B-mScarlet) and actin (mNeon-ACTB) in MDA-MB 231 cells. Scale bars are 10 μm .

To overcome this limitation, Noise2Noise was developed⁴. Noise2Noise can be trained exclusively on pairs of noisy images without any access to ground truth data. It is especially useful in biological imaging where, often, imaging trade-offs dictate that ground truth data cannot ever be obtained.

However, paired noisy images are not always easy to obtain, so there was interest in developing methods that could denoise on unpaired training sets of noisy images from some desired domain. The first method capable of this without having sensitive hyperparameters was Noise2Void⁶. Noise2Void works by training the network to learn a mapping from the noisy image back to itself, masking the centre of each receptive field so as to avoid learning the identity.

This basic model of masking the input is known as a blind-spot network, and was heavily refined and expanded upon in ref.¹⁹ and much more recently applied in BP-AIDE²⁰ in a manner that is specifically tailored to Gaussian–Poisson noise. In ref.²¹ they demonstrate a retooled version of BP-AIDE with much faster inference time.

A recently developed alternative to blind-spot networks is Neighbor2Neighbor⁹ which underlies the method we present in this paper. Neighbor2Neighbor learns to map adjacent pixels in the image to one-another, with the idea being that, except in the most highly

dynamic regions of the image, adjacent pixels tend to have a similar underlying signal.

Recorrputed-to-Recorrputed²² is another recent denoiser. Recorrputed-to-Recorrputed attempts to corrupt single noisy images into noisy image pairs, and then apply a Noise2Noise-like network. Recorrputed-to-Recorrputed is not blind, and requires an estimate of the underlying noise variance and also contains a sensitive ‘coefficient of recorrputation’ parameter.

Ultimately, all methods listed in this section require a representative training set of noisy images to train on before being applied. In the next section we describe methods that were specifically developed for denoising single noisy images without a training set.

Zero-shot methods

The first method that directly applied itself to the task of blind zero-shot denoising is Noise2Self⁷. Noise2Self is a very similar method to Noise2Void that achieves slightly better performance, and includes a very thorough mathematical justification for the principles underlying the success of masking-based denoising techniques.

Self2Self⁸ was the first blind zero-shot method whose performance approaches fully trained methods. Self2Self is a blind-spot method,

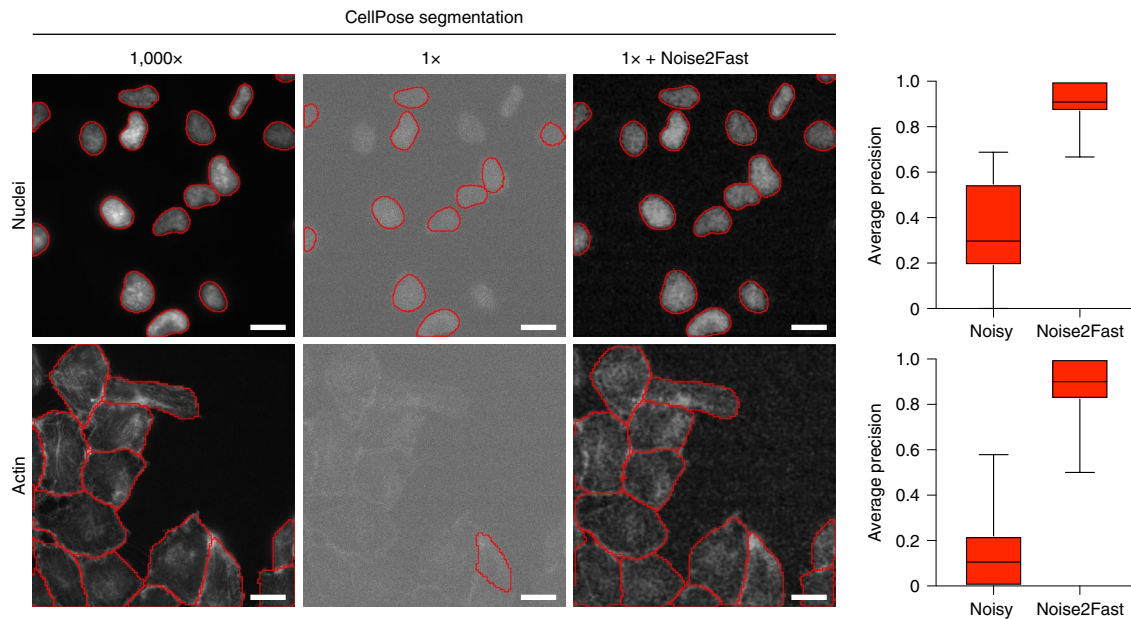


Fig. 4 | We demonstrate that Noise2Fast significantly improves downstream segmentation with CellPose (a generalist segmentation tool) on our data. We show that by using Noise2Fast on low exposure 1× (0.1 milliseconds) images, we can achieve generalist segmentation results that nearly match that of our high

exposure 1,000× (100 milliseconds) pseudo ground truth images. We tested on four crops extracted from each of three fields of view, for a total of eleven images in each dataset (one image was excluded because it contained nothing). All images were taken from the same sample. Scale bars are 10 μm .

however instead of replacing masked pixels, it ignores them altogether by using partial convolutions^{23,24}. Self2Self also introduces the innovative step of adding dropout and averaging across multiple runs of the same image. However, this comes at a high computational cost.

Non-blind zero-shot methods

BM3D²⁵ is one of the gold standards for pure Gaussian noise. It works by unfolding the image into interleaved square patches, clustering those patches based on similarity, and then filtering them before reconstructing the image. BM3D, however, is not blind and takes, as a parameter, an estimate of the standard deviation of the underlying noise. Moreover, BM3D does not work on Poisson noise.

A much more recent learning-based method is Deep Image Prior (DIP)²⁶. DIP works by taking a neural network with randomly initialized weights, and training to reconstruct the noisy image. Similar to Noise2Noise, it will fail to learn the underlying noise (at least at first) and instead learn to output the signal. DIP is highly sensitive to the number of iterations, and will quickly overfit if trained too long, for this reason it is not completely practical as a blind denoiser. For our experiments, we force it to be blind by using a fixed iteration number, however, the results it attains are far below what a non-blind version of this algorithm can reach.

Since it is easy to confuse the various different types of unsupervised denoiser, we have included a chart in Extended Data Fig. 2 to clarify the distinctions.

Results

Accuracy

The benchmarking of reference datasets was carried out using a single laptop GPU (RTX 5000 mobile GPU) to better approximate the modest (although still powerful) computational capabilities of the average end user. However, because of the massive amount of time required to test Self2Self on 68 images under these constraints, we rely on their previously published accuracy measurement for comparison and estimate time per image using a random sample of five images for this dataset only. On all other datasets (Set12 and Confocal), we run Self2Self on the entire set to obtain accuracy and speed.

On synthetic Gaussian noise our method outperforms everything except Self2Self, which beats us by 0.6–1.0 PSNR across Set12 and BSD68 (Table 1 and Fig. 2). We also tested our method on confocal microscopy images, where again we are slightly less accurate than Self2Self, but outperform everything else. Visual comparison of the results (Fig. 2b) indicate that Noise2Fast appears to smooth the image less than the other methods, creating a more textured look. But overall, every method performed very similarly on the confocal microscopy dataset, except for DIP, which likely needed more iterations to converge, and Neighbor2Neighbor, which seems to not really be suited to zero-shot denoising (nor was it ever intended to be).

For comprehensiveness, we also do pure accuracy comparisons for a myriad of other methods in Extended Data Fig. 3, including Noisier2Noise²⁷ and SURE^{28,29} where compared methods have access to varying degrees of additional information ranging from an estimate of the noise level (BM3D) to a full representative noisy/clean dataset (DnCNN).

Speed

We next sought to determine how fast our method is compared to other algorithms. Our results show that Noise2Fast is considerably faster than all tested methods, an average of 200 times faster than Self2Self the only method that exceeds us in accuracy (Table 1 and Fig. 2).

Because ‘speed’ is just a reflection of the maximum number of iterations we allow each method to run (a parameter we borrow from their published code where possible), we also compared the accuracy of each method if we set the maximum number of iterations so that each program only runs for as long as Noise2Fast takes to fully denoise the image (Extended Data Fig. 4). In this case, it is easy to see that no competitor even approaches the accuracy Noise2Fast can achieve in such a short amount of time.

Biological data

Next we determined the speed of Noise2Fast on larger image datasets of both fixed and live cells acquired on our imaging systems. For this, MDA-MB 231 cells were fixed and either stained for actin and DNA or endogenously tagged with H3-3B-mScarlet and mNeon-ACTB (Methods). The performance was compared using two different imaging

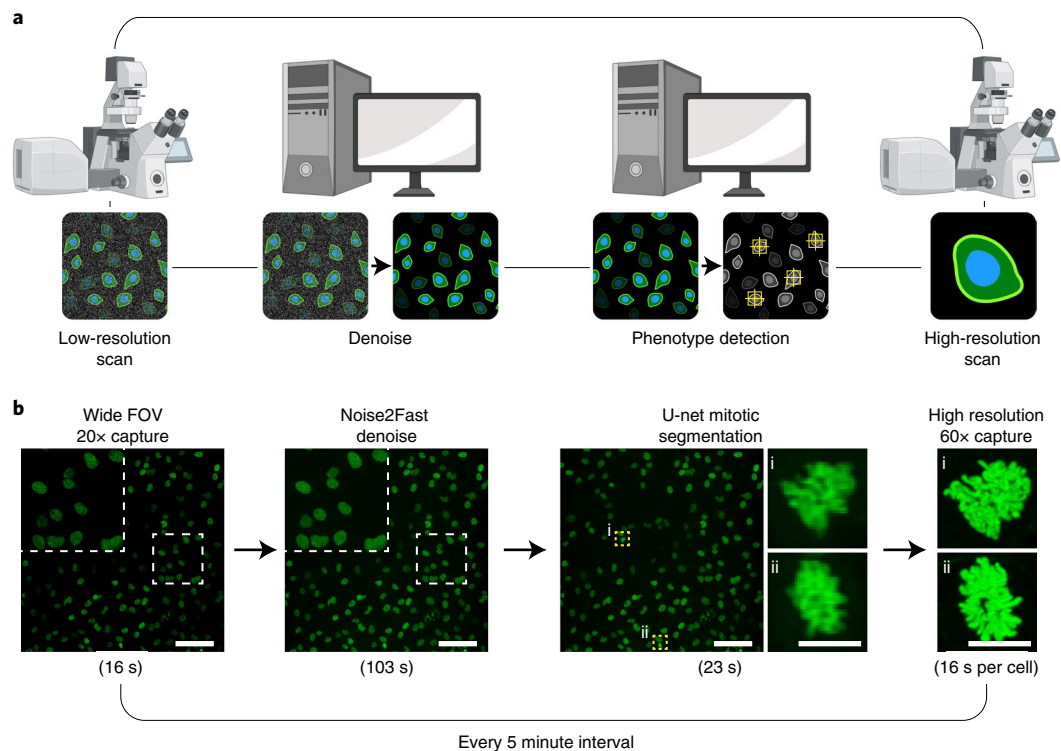


Fig. 5 | Application of Noise2Fast. a, Proposed automated pipeline where the microscope detects a transient phenotype of interest, and both zooms in on and images any cells displaying that phenotype. **b**, Such a pipeline for detecting mitotic cells. Every 5 minutes, the microscope captures a wide field of view 20×

image, denoises using Noise2Fast, scans this large field of view for any mitotic cells and then zooms in on them to image at 60×. Under these time constraints, we can detect up to eight mitoses in each 5 minute interval. Scale bars are 100 μm. Created with [BioRender.com](https://www.biorender.com).

modalities: epifluorescence for the fixed cells and resonance scanning confocal microscopy for the live cells. Based on the linearity of the intensity measurements of our imaging system, our results indicate that we can achieve relatively clear images while exposing our images to 400-fold less light (Fig. 3). Although Self2Self achieves similar, if not slightly better, results, processing time was significantly longer, more specifically Self2Self required 596 core days versus 0.7 for our method to process the video in Fig. 3b on a Tesla V100.

Effect on downstream analysis

We next sought to determine if Noise2Fast improves downstream segmentation tasks. Towards this, in Fig. 4 we show how denoising impacts segmentation results using CellPose³⁰. We chose CellPose for this comparison because it is commonly used and it pairs well with Noise2Fast since both are generalist. Our results show a clear improvement in segmentation accuracy when the image is first denoised using Noise2Fast, as quantified using the average precision metric used in refs.^{31,32}. Our results highlight the need to incorporate denoising as a first step in image analysis pipelines when noise levels are high, and in this case we show that Noise2Fast can effectively serve in this role. For a more thorough investigation of how denoising improves segmentation, see ref.³¹.

Application

We show that our method can be integrated into time-sensitive automated microscopy pipelines. In particular, we performed an experiment where the microscope monitors a large field of view (FOV) at 20× (containing about 40 cells) over 8 hours (Fig. 5). Whenever the microscope detects a mitotic cell in the large FOV, it zooms in on that cell and images at 60×. Mitotic detection is achieved by first using CellPose to segment the cells, and then distinguishes mitotic from interphase cells by using a cut-off on the standard deviation of the pixel intensity value (anything over 300 is deemed mitotic). This experiment allows

us to scrutinize phenotypes that manifest prominently during mitosis, such as centrosome amplification, without sacrificing throughput.

Conclusion

We presented Noise2Fast, a blind zero-shot denoiser that rapidly converges to accurate results using only the input image to train on. Our key innovation is building a small discrete training set based on checkerboard downsampling that enables our network to quickly converge. We can monitor the progress training using original noisy image as validation. The accuracy of our method surpasses all but one tested blind zero-shot denoising methods, namely Self2Self, however Self2Self takes an average of 200 times longer to run and is therefore impractical in real-time situations, such as in live-cell experiments where the microscope must act upon transient information.

To this end, we successfully integrated Noise2Fast into a live-cell analysis pipeline where the microscope detects and zooms in on any mitotic cells. This example shows that our speed gain is not only an asset, but a necessity to these kinds of analyses. Additionally, from a more theoretical perspective, we believe the observed superiority of checkerboard downsampling over traditional 2×2 downsampling is noteworthy, and the implications this has for full dataset-based denoising methods such as Neighbor2Neighbor might be a worthwhile subject of future research.

Methods

Noise2Fast implementation details

Here we outline the specifics of our neural network and training scheme, giving the implementation details of the process outlined earlier. We note that for all zero-shot denoisers we test, noise generation is carried out separately from denoising. That is to say, each method is only exposed to one single noisy instantiation of one single image during training.

For our neural network, we use a simple CNN architecture which we explain briefly here, and illustrate in Fig. 1b. We start by performing two 32 channel 3×3 convolutions with ReLU activation. We repeat this step three more times, each time doubling the number of channels. In the final step, we do 1×1 convolution followed by sigmoid activation.

In our initial testing we found that this much simpler architecture outperformed the classical U-net architecture used in the original Noise2Noise paper⁴. Although the results are not that sensitive to the number of hidden layers, we do find a noticeable, albeit small, drop in performance as we add more to our current model. A possible reason for this is that it causes our network to overfit the data much too quickly. This architecture is similar in its simplicity to DnCNN, one major difference being our lack of batch normalization.

The main novelty of our method is how we train it. Consider a 2D image $\mathbf{x} \in \mathbb{R}^{m \times n}$. Recall from the theoretical background that we can divide our image in two by using checkerboard downsampling. By taking the even or odd pixels and squeezing them up to fill in the spaces, as depicted in Fig. 1, we can generate two downsampled $m \times \frac{1}{2}n$ images

$$\mathbf{x}_{\text{even}}(i, j) = \mathbf{x}(i, 2j + (i \bmod 2)), \quad (16)$$

$$\mathbf{x}_{\text{odd}}(i, j) = \mathbf{x}(i, 2j + (i \bmod 2) + 1). \quad (17)$$

We can call these the ‘up’ checkerboard downsamples. Notice that we can also squeeze the pixels left to generate two $\frac{1}{2}m \times n$ images

$$\mathbf{x}'(i, j) = \mathbf{x}(2i + (j \bmod 2), j), \quad (18)$$

$$\mathbf{x}'(i, j) = \mathbf{x}(2i + (j \bmod 2) + 1, j). \quad (19)$$

Giving us the ‘left’ checkerboard downsamples. Using these we construct a four-image-pair training set (see Fig. 1c for an overview of our training scheme). We then feed this training data one-by-one into our neural network (batch size = 1). At each iteration we compute the binary cross-entropy (BCE) loss between the target and the output of our neural network, and adjust our weights using the Adam optimizer^{33,34} with learning rate set to 0.001.

When we downsample our image we fundamentally distort the pixel lattice and the relationship between adjacent pixels. For example, in a normal image, suppose each pixel is 1×1 μm. Then the pixel immediately above will be 1 μm away, and the pixel immediately to the right will also be 1 μm away. In the case of a checkerboard downsampled image, for example \mathbf{x}_{even} in Fig. 1, depending on where you are the relationship might be something like: Up: 2 μm, Left: $\sqrt{2}$ μm. The relationship becomes even more complex as you move further away and the net effect is a serious disruption of the underlying relationship between a pixel and neighbours. We note that this is different from adopting a checkerboard masking scheme and training Noise2Void, where the relationship and adjacency between pixels is preserved. And even Neighbor2Neighbor at least preserves ‘ratios’ in the sense that the vertical and horizontal scale are divided by a common number, and these proportions do not vary from pixel to pixel.

This has one important effect: while blind-spot methods based on masking get more and more accurate over time without overfitting because they are not given sufficient information to overfit the data, our method performs more like DIP, where accuracy reaches a maximum very quickly, before it starts to plummet as it ultimately overfits the data, this effect happens particularly quickly on our small four-image training set. However, since our distorted downsampled data looks nothing like the original noisy image at a local level, we observe that if we train our four-image checkerboard downsampled set, while we do inevitably overfit our small training set, the original noisy image is basically unseen data for all intents and purposes. Therefore, we can actually use this image as a validation set.

More specifically, after each iteration we monitor how our neural network maps our original noisy image \mathbf{x} to itself, that is how close $f(\mathbf{x})$ is to the identity mapping on \mathbf{x} . Our observation is that the optimal time to ‘stop’ training is quite close to the point where the output of the neural network, when applied to original noisy image, most resembles itself. We can validate in this way only because our training set images look so vastly different at a local level than the image we are trying to denoise. In Extended Data Fig. 5 we illustrate how this validation strategy works by comparing over time known ground truth PSNR to the validation PSNR determined by comparing our output to the original noisy image. As can be seen in this figure, the two lines move together and achieve their peaks at roughly the same time. Ultimately, our ability to validate in this way results in a method that converges both quickly and accurately.

Compared datasets

For blind Gaussian denoising we use the greyscale BSD68³⁵ dataset, as was used in ref. ⁶ and a multitude of other denoising papers. BSD68 consists of 68 clear 481×321 photographs to which we add synthetic Gaussian noise. However, to show the effect of spatial resolution on speed and performance, we additionally tested the methods on Set12 which contains a mixture of 256×256 and 512×512 images.

For performance on real-world confocal microscopy, we used a subset of the confocal microscopy images in Fluorescent Microscopy Dataset (FMD)² that we refer to as ‘Confocal’. This dataset contains, among other things, images of biological materials such as cells, zebrafish and mouse brain tissues acquired using commercial confocal microscopes. As described in their paper, ground truth values are estimated by averaging together all 60,000 noisy images in a given set.

Compared methods

We compare denoising and speed performance against five other blind zero-shot denoisers: Noise2Self⁷, Noise2Void⁶, Self2Self⁸, Neighbor2Neighbor⁹ and DIP²⁶. Not all of these methods were originally designed for zero-shot denoising. We will describe how we configured each of these methods in turn, we adhere to published code as much as possible.

Self2Self. For Self2Self we use the default published settings of 150,000 iterations and a learning rate of 1×10^{-4} . We standardize our images differently than Self2Self and some of these other methods. For example, we do not clip our input noisy data [0, 255] at any point. To account for this difference, we have rewritten the dataloaders for Self2Self and other methods to ensure consistency of comparison.

Noise2Self. For Noise2Self the only change we make from their published single-shot denoising notebook is to increase the number of iterations from 500 to 20,000, as we found that 500 iterations were not nearly enough to achieve good results on these datasets.

Noise2Void. For Noise2Void we found that their ImageJ plugin worked much better than their GitHub code for zero-shot denoising. We therefore used the ImageJ version for benchmarking purposes, which is why our results on this method deviate so much from previous publications. We used a patch size of 64 with 100 epochs and 100 steps per epoch, a batch size of 16 per step, and a neighbourhood radius of 5.

DIP. If we fix the maximum number of iterations, DIP becomes a blind denoiser. However, as noted in ref. ⁸, it performs better as a non-blind denoiser. For comparison purposes however, we will set the maximum number of iterations at 3,000, as the authors of DIP have done in their example code on GitHub. This turns it into a blind single-shot denoiser, fully comparable in scope to our method.

Neighbor2Neighbor. For Neighbor2Neighbor we used the adaptation of the code found here: <https://github.com/neeraj3029/Ne2Ne-Image-Denoising>. We adapted the script to zero-shot denoising

and attempted in good faith to optimize for the task as best we could, however, we found that the results were inconsistent. We believe that this method is probably best suited to datasets as the authors intended and not single images. We include these results only to illustrate the need to change Neighbor2Neighbor in order to achieve fast and accurate zero-shot denoising results, as we have done in this paper. We do not believe our results are a fair illustration of the power of Neighbor2Neighbor when applied to the tasks it was designed for and we have therefore excluded it from our visual illustrations. We used a learning rate of 0.0003 and trained for 100 epochs, as suggested in their paper for synthetic datasets.

Fluorescence microscopy images

For fixed immunofluorescence microscopy, RPE-1 cells were fixed with 4% paraformaldehyde at room temperature for 10 min. The cells were then blocked with a blocking buffer (5% BSA and 0.5% Triton X-100 in PBS) for 30 min. Cells were washed with PBS and subsequently incubated with phalloidin-Alexa488 (Molecular Probes) and DAPI in blocking solution for 1 hour. After a final wash with PBS, the coverslips were mounted on glass slides by inverting them onto mounting solution (ProLong Gold antifade; Molecular Probes). For the fixed imaging in Fig. 3a, single Z slices of cells were imaged using Nikon Ti2E/CREST X-Light V2 LFOV25 spinning disk confocal microscope in widefield mode using a 60×/1.4 NA oil-immersion Plan-Apochromat lambda objective. The microscope was outfitted with a Photometrics Prime95B 25 mm FOV ultra-high sensitivity sCMOS camera and images were captured with no binning using the full 25 mm diagonal FOV area at 1,608 px by 1,608 px with a bit depth of 16 bit. After capture, 500 px by 500 px areas were cropped and used as our input dataset. For live imaging in Fig. 3b, endogenously tagged MDA-MB 231 cells were seeded in Nunc Lab-Tek Chamber Slides and imaged on the Nikon Ti2E/AIR-HD25 scanning confocal microscope with temperature and CO₂ control, using a 40×/1.15 NA water-immersion objective Apochromat lambda S objective. High-speed image acquisition was carried out with the resonance scan head with 2× averaging at 1,024 px by 1,024 px. Full volumes of cells were captured ($Z_{\text{total}} = 20 \mu\text{m}$, $Z_{\text{interval}} = 0.5 \mu\text{m}$) every 5 minutes for 24 hours. For Fig. 4, single Z slices of cells were imaged using Nikon Ti2E/CREST X-Light V2 LFOV25 spinning disk confocal microscope in widefield mode using a 60×/1.4 NA oil-immersion Plan-Apochromat lambda objective. The microscope was outfitted with a Photometrics Prime95B 25 mm FOV ultra-high sensitivity sCMOS camera and images were captured at two different exposures (0.1 and 100 ms) with no binning using the full 25 mm diagonal FOV area at 1,608 px by 1,608 px with a bit depth of 16 bit. After capture, 500 px by 500 px areas were cropped (this you specify to your cropping in this figure) and used as our input dataset. Images were denoised as individual Z-slices and max projected. All are displayed with auto scaled LUTs.

Ablation study

For our ablation study, we compare three different refinements of the model. First, we replace our simple neural network with a U-net architecture, which is the standard network used in Self2Self and Noise2Void. Again, our performance drops (Extended Data Fig. 6, U-net). Also, using known ground truth values, we manually subtract out the $s_{\text{odd}} - s_{\text{even}}$ term in equation (12) and show that this has virtually no impact on our denoising results, hence this term is not having a significant impact on our algorithm (Extended Data Fig. 6, Exact). Finally, we test how well Noise2Fast works if instead of applying it to the original noisy image, we apply it to the checkerboard downsampled images and reassemble them into the full-sized image (Extended Data Fig. 6, Split).

We also investigate what happens if we replace our unusual checkerboard downsampling with a more conventional downsample where we divide our image into 2×2 blocks, as used in Neighbor2Neighbor and also 3×3 blocks for fourfold and ninefold downsampling, respectively. This has the advantage making our training set consist of even smaller

images to further reduce computation time. We test this on the confocal dataset, and as can be seen in Extended Data Fig. 7, fourfold downsampling doubles speed at only a small drop in accuracy, making this perhaps an attractive solution to those looking for even more speed. Ninefold sampling on the other hand only increases speed marginally and with a much steeper drop in accuracy. The diminishing returns in speed gain are likely a result of the neural network requiring longer to converge when there is less training data per iteration.

We also compare the effect our small architecture has on our results by adding the DIP hourglass architecture to Noise2Fast (Extended Data Fig. 8), and also by inserting our architecture into Self2Self (Extended Data Fig. 9) and running it on the parrot image from Set12.

Source of cell lines

MDA-MB 231 cell line was a gift from R. S. Kerbel (Sunnybrook Health Sciences Centre, Toronto, Canada) and cultured at 37 °C in a humidified environment containing 5% CO₂. MDA-MB 231 cells were grown in Roswell Park Memorial Institute (RPMI) 1640 medium (Life Technologies) supplemented with 10% fetal bovine serum (FBS). RPE-1 (CRL-4000) cell line was acquired from the American Type Culture Collection (ATCC) and grown in Dulbecco's Modified Eagle Medium/Nutrient Mixture F12 (DMEM/F12 1:1; Life Technologies) supplemented with 10% FBS.

To generate our endogenous fluorescent cell lines, CRISPR-Cas9 was paired with a repair construct to insert sequences encoding fluorescent proteins via homology-directed recombination into the N- and C-terminus of ACTB and H3-3B, respectively. Briefly, sgRNAs targeting the N-terminus of ACTB (GCCGTTGTCGACGACGAGCGCGG) and the C-terminus of H3-3B (CAGTTGGCTCGCCGGATACGGGG) were cloned into a pX330 plasmid (Addgene plasmid #42230) following the Zhang protocol³⁶. To generate the repair constructs, 10,000 bp of homologous genomic sequence surrounding the sgRNA cut site of ACTB and H3-3B was amplified from the genome of RPE-1 cells and subsequently Gibson (M510AA; NEB) assembled to flank a cassette containing mNeon or mScarlet followed by a 2A peptide into a puromycin or blasticidin resistance gene. Cells were co-transfected with a sgRNA-cloned pX330 plasmid and its matching repair construct using Lipofectamin3000 (Invitrogen) at a 1:1 ratio. An editing period of 72 hours was allotted prior to selection with puromycin (2 mg ml⁻¹) or blasticidin (10 mg ml⁻¹) to cull non-integrated cells. Knock-in positive cells were subsequently FACS sorted and inspected via fluorescence microscopy.

Reporting summary

Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

Benchmarking datasets along with code and reproducibility instructions for Fig. 2 are available on our GitHub (<https://github.com/pelletierlab/Noise2Fast>). Note that all speed benchmarks were performed on an RTX 5000 mobile GPU, and therefore results may vary according to GPU used. Source input and output images used to make the graphs in Fig. 4 are publicly available on our GitHub (Noise2Fast/Fig5Data). Source data for Fig. 3 is available on our GitHub as well (Noise2Fast/livecells). Figure 1 is a conceptual illustration and does not make use of any datasets, however the image we use to illustrate checkerboard downsampling is a crop of an image available on our GitHub (Noise2Fast/BSD68/19.tif). The minimum dataset for the experiment illustrated in Fig. 5 is publicly available on our GitHub (Noise2Fast/Fig6Data) and the full source data is available on Zenodo (<https://doi.org/10.5281/zenodo.6949784>).

Code availability

Our code, along with reproducibility instructions, is publicly available at <https://github.com/pelletierlab/Noise2Fast>. It is also available on codeocean, <https://doi.org/10.24433/CO.7967850.v137>.

References

- Goyal, B. et al. Image denoising review: from classical to state-of-the-art approaches. *Inf. Fusion* **55**, 220–244 (2020).
- Zhang, Y. et al. A Poisson-Gaussian denoising dataset with real fluorescence microscopy images. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 11710–11718 (2019).
- Zhang, K., Zuo, W., Chen, Y., Meng, D. & Zhang, L. Beyond a Gaussian denoiser: residual learning of deep cnn for image denoising. *IEEE Trans. Image Process.* **26**, 3142–3155 (2017).
- Lehtinen, J. et al. Noise2Noise: Learning image restoration without clean data. In *Proc. 35th International Conference on Machine Learning* (eds Dy, J. & Krause, A.) Vol. 80, 2965–2974 (PMLR, 2018).
- Weigert, M. et al. Content-aware image restoration: pushing the limits of fluorescence microscopy. *Nat. Methods* **15**, 1090–1097 (2018).
- Krull, A., Buchholz, T.-O. & Jug, F. Noise2void - learning denoising from single noisy images. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019).
- Batson, J. & Royer, L. Noise2Self: Blind denoising by self-supervision. In *Proc. 36th International Conference on Machine Learning* (eds Chaudhuri, K. & Salakhutdinov, R.) Vol. 97, 524–533 (PMLR, 2019).
- Quan, Y., Chen, M., Pang, T. & Ji, H. Self2self with dropout: Learning self-supervised denoising from single image. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020).
- Huang, T., Li, S., Jia, X., Lu, H. & Liu, J. Neighbor2neighbor: self-supervised denoising from single noisy images. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* 14781–14790 (2021).
- Zontak, M. & Irani, M. Internal statistics of a single natural image. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 977–984 (2011).
- Glasner, D., Bagon, S. & Irani, M. Super-resolution from a single image. In *2009 IEEE 12th International Conference on Computer Vision* 349–356 (2009).
- Zhang, Y., Ling, F., Li, X. & Du, Y. Super-resolution land cover mapping using multiscale self-similarity redundancy. *IEEE J. Sel. Topics Appl. Earth Obs. Remote Sens.* **8**, 5130–5145 (2015).
- Hasle, N. et al. High-throughput, microscope-based sorting to dissect cellular heterogeneity. *Mol. Syst. Biol.* **16**, e9442 (2020).
- Kanfer, G. et al. Image-based pooled whole-genome CRISPRi screening for subcellular phenotypes. *J. Cell Biol.* **220**, e202006180 (2021).
- Jain, V. & Seung, S. Natural image denoising with convolutional networks. In *Adv. Neural Information Processing Systems* (eds Koller, D. et al.) Vol. 21, 769–776 (Curran Associates, 2009).
- Mao, X., Shen, C. & Yang, Y.-B. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Adv. Neural Information Processing Systems* (eds Lee, D. et al.) Vol. 29, 2802–2810 (Curran Associates, 2016).
- Zhang, K., Zuo, W. & Zhang, L. FFDNet: toward a fast and flexible solution for cnn-based image denoising. *IEEE Trans. Image Process.* **27**, 4608–4622 (2018).
- Shi, W. et al. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016).
- Laine, S., Karras, T., Lehtinen, J. & Aila, T. High-quality self-supervised deep image denoising. In *Adv. Neural Information Processing Systems* (eds Wallach, H. et al.) Vol. 32 (Curran Associates, Inc., 2019).
- Byun, J. & Moon, T. Learning blind pixelwise affine image denoiser with single noisy images. *IEEE Signal Process. Lett.* **27**, 1105–1109 (2020).
- Byun, J., Cha, S. & Moon, T. FBI-Denoiser: fast blind image denoiser for poisson-gaussian noise. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* 5768–5777 (2021).
- Pang, T., Zheng, H., Quan, Y. & Ji, H. Recorruped-to-recorruped: unsupervised deep learning for image denoising. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* 2043–2052 (2021).
- Liu, G. et al. Image Inpainting for Irregular Holes Using Partial Convolutions. In *The European Conference on Computer Vision (ECCV)* (2018).
- Buades, A., Coll, B. & Morel, J.-M. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (IEEE)*.
- Dabov, K., Foi, A., Katkovnik, V. & Egiazarian, K. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Trans. Image Process.* **16**, 2080–2095 (2007).
- Ulyanov, D., Vedaldi, A. & Lempitsky, V. Deep image prior. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (2018).
- Moran, N., Schmidt, D., Zhong, Y. & Coady, P. Noisier2Noise: Learning to Denoise From Unpaired Noisy Data. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).
- Soltanayev, S., Chun, S.-Y. & Zhong, Y. Training deep learning based denoisers without ground truth data. In *Adv. Neural Information Processing Systems* Vol. 31 (2018).
- Lequyer, J., Philip, R., Sharma, A., Hsu, W.-H. & Pelletier, L. Noise2Fast a fast blind zero-shot denoiser. *GitHub* <https://doi.org/10.5281/ZENODO.655691> (2021).
- Stringer, C., Wang, T., Michaelos, M. & Pachitariu, M. Cellpose: a generalist algorithm for cellular segmentation. *Nat. Methods* **18**, 100–106 (2021).
- Buchholz, T., Prakash, M., Krull, A. & Jug, F. DenoiSeg: Joint Denoising and Segmentation. In *Computer Vision – ECCV 2020 Workshops* (Springer, 2020).
- Yan, X. et al. High-content imaging-based pooled CRISPR screens in mammalian cells. *J. Cell Biol.* **220**, e202008158 (2021).
- Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. In *Proc. 3rd International Conference on Learning Representations (ICLR)* (2015).
- Lin, T.-Y., Goyal, P., Girshick, R., He, K., Dollár, P. & Lin, T.-Y. Focal Loss for Dense Object Detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 42, 318–327 (2020).
- Martin, D., Fowlkes, C., Tal, D. & Malik, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th International Conf. Computer Vision* Vol. 2, 416–423 (2001).
- Cong, L. et al. Multiplex genome engineering using CRISPR/Cas systems. *Science* **339**, 819–823 (2013).
- Lequyer, J., Philip, R., Sharma, A., Hsu, W.-H. & Pelletier, L. Noise2Fast: a fast self-supervised single image blind denoiser. <https://codeocean.com/capsule/2050494/tree/v1> (2022).

Acknowledgements

We thank members of the Pelletier Lab for their scientific feedback during the project. We are grateful to S. Prosser, J. Kim and J. Tkach for proofreading the manuscript. J.L. and R.P. were funded in part by the Lunenfeld Tanenbaum Research Institute Studentships at Sinai Health System. The remainder of this work was funded by CIHR Foundation (FDN # 167279) and Krembil Foundation grants to L.P., which were used to fund J.L., W.-H.H., A.S. and R.P.; L.P. is a Tier 1 Canada Research Chair in Centrosome Biogenesis and Function. The Network Biology Collaborative Centre at the LTRI is supported by the Canada Foundation for Innovation, the Ontario Government, and Genome Canada and Ontario Genomics (OGI-139).

Author contributions

J.L. conceived of and coded the presented algorithm and wrote the manuscript. R.P. imaged the biological structures presented in Fig. 3, helped make Figs. 3 and 5, and assisted in writing the manuscript. A.S. engineered the cells that were imaged by R.P.; W.-H.H. extensively edited and reworked the figures and data presentation and assisted in writing the manuscript. L.P. encouraged J.L. to investigate denoising, supervised the findings of this work, assisted in writing the manuscript and funded the project. All authors were presented with and approved of the manuscript before submission.

Competing interests

The authors declare no competing interests.

Additional information

Extended data is available for this paper at <https://doi.org/10.1038/s42256-022-00547-8>.

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s42256-022-00547-8>.

Correspondence and requests for materials should be addressed to Laurence Pelletier.

Peer review information *Nature Machine Intelligence* thanks Hui Ji, Alexander Krull, and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2022

Image	Itself	Chequerboard Downsamplings of Itself				Other Set12 Images		
	Image 11	Odd-Left	Even-Left	Odd-Up	Even-Up	Image 12	Image 9	Image 10
Average number of similar patches with Image 11	301.4	202.3	192.7	151.9	155.6	1.4	21.7	25.3

Extended Data Fig. 1 | Patch similarity in chequerboard downsamplings. We compare the number of similar patches that exist between Image 11 from Set12, its chequerboard downsamplings and other Set12 images.

Types of Unsupervised Denoiser

Type of Denoiser	Access to Representative Training Set (Unpaired)	User-Supplied Knowledge of Noise Standard Deviation	Examples
Non-blind single-shot	✓	✓	Recorrputed-to-Recorrputed
Blind single-shot	✓	✗	Noise2Void*, Noise2Self*, Neighbor2Neighbor*
Non-blind zero-shot	✗	✓	BM3D
Blind zero-shot	✗	✗	Noise2Fast, Self2Self, DIP3000, Noise2Self (Single) [†] , Noise2Void (Single) [†]

*When trained on a representative dataset

[†]When trained on a single image

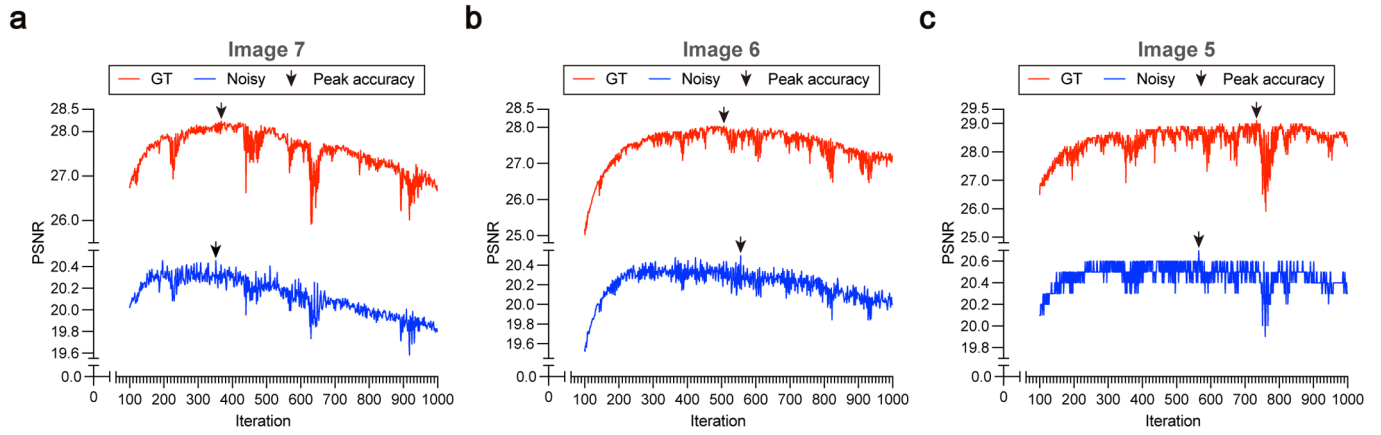
Extended Data Fig. 2 | Comparing different types of unsupervised denoiser. We classify the most popular unsupervised denoisers based upon how much information they have access to prior to inference.

Dataset	σ	Blind zero-shot						Non-blind zero-shot	Blind single-shot		Non-blind single-shot	Supervised
		Noise2Self (single) PSNR/SSIM	Noise2Void (single) PSNR/SSIM	DIP3000 PSNR/SSIM	Self2Self PSNR/SSIM	Neighbor2 Neighbor (single) PSNR/SSIM	Noise2Fast PSNR/SSIM	BM3D PSNR/SSIM	SURE PSNR/SSIM	Noisier2Noise PSNR/SSIM	Recorruprted-to-Recorruprted PSNR/SSIM	DnCNN PSNR/SSIM
Set12	15	30.19/8.71	30.04/8.47	28.51/8.04	32.17/8.89	27.97/7.87	31.10/8.71	32.39/8.96	32.45/8.93	32.16/8.83	32.50/8.95	32.86/9.03
	25	28.23/8.12	28.41/7.80	26.47/7.07	29.88/8.42	26.23/6.87	29.05/8.22	29.97/8.51	30.00/8.46	29.66/8.41	30.04/8.50	30.42/8.62
	50	24.78/6.78	25.70/7.20	21.19/4.39	26.34/7.34	23.43/5.32	25.82/7.23	26.75/7.67	26.37/7.36	26.07/7.31	26.73/7.59	27.15/7.83
BSD68	25	27.24/7.72	26.66/7.31	25.74/6.85	28.70/8.03	25.68/6.92	28.12/7.89	28.62/8.04	28.98/8.20	28.59/8.11	28.93/8.14	29.24/8.28
	50	24.43/6.41	24.50/6.05	21.29/4.42	25.92/6.99	23.59/5.45	25.23/6.70	25.67/6.87	25.96/6.88	25.66/6.87	25.93/6.94	26.22/7.19

Extended Data Fig. 3 | Additional accuracy comparisons on Gaussian noise. Comparing the accuracy performance of Noise2Fast on Gaussian noise, against a myriad of other methods.

Dataset	σ	Noise2Self (single)	Noise2Void (single)	DIP3000	Neighbor2Neighbor (single)	Self2Self	Noise2Fast
		PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM
Set12	15	26.68/7.91	26.09/7.30	28.20/8.01	25.01/6.96	27.03/7.99	31.10/8.71
	25	25.72/7.20	24.48/6.47	26.53/7.35	24.07/5.83	25.83/7.23	29.05/8.22
	35	24.10/6.04	22.01/5.00	24.95/6.50	21.88/4.78	25.10/6.52	27.57/7.81
	50	23.32/5.71	21.98/4.99	22.28/5.04	21.23/3.94	24.21/6.10	25.82/7.23
BSD68	25	25.48/7.03	22.91/6.53	25.62/6.91	22.20/5.47	25.32/6.91	28.12/7.89
	50	23.56/6.01	22.42/4.91	22.08/4.83	20.62/3.65	23.83/6.33	25.23/6.70
Confocal	-	36.18/9.28	32.54/8.26	35.22/9.07	10.50/1.72	28.01/6.47	36.61/9.33

Extended Data Fig. 4 | Comparing accuracy when we force methods to be fast. We show the accuracy reached by each method, by the time Noise2Fast has completed its denoising.



Extended Data Fig. 5 | Illustrating our validation strategy on Set12 images. PSNR over time of Noise2Fast on images 5, 6 and 7 from Set12 with $\sigma = 25$ Gaussian noise. PSNR is calculated by comparing the output of Noise2Fast to both the original ground truth image (red), and to the original noisy image (blue).

σ	Normal	Unet	Exact	Split
25	28.12	27.76	28.12	24.81
50	25.23	25.01	25.24	23.58

Extended Data Fig. 6 | Comparing refinements to Noise2Fast. Comparing different possible refinements to Noise2Fast on BSD68 with $\sigma = 25$ Gaussian noise.

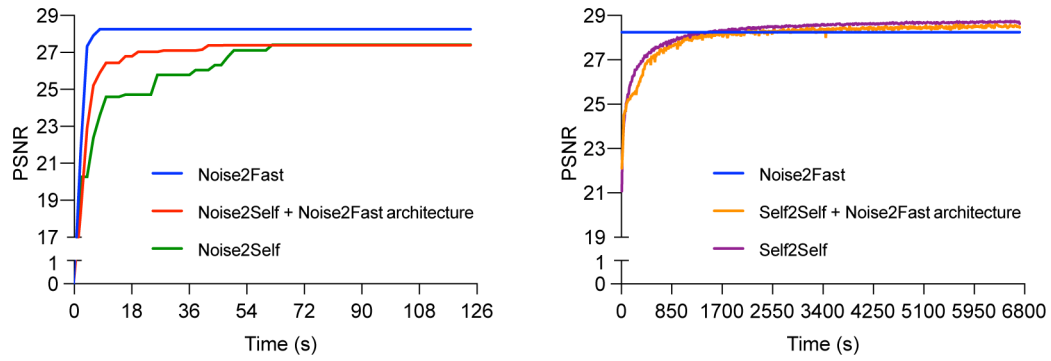
	2-fold (Normal)	4-fold	9-fold
PSNR	36.61	36.45	35.66
Time/Image	56s	26s	22s

Extended Data Fig. 7 | Comparing alternative downsampling strategies. Comparing alternatives to checkerboard downsampling on the Confocal dataset.

Noise2Fast		Noise2Fast + DIP architecture		Noise2Fast + DIP architecture, halted using GT	
PSNR/SSIM	Time/Image	PSNR/SSIM	Time/Image	PSNR/SSIM	Time/Image
31.10/8.71	22s	30.23/8.53	11s	30.24/8.53	13s
29.05/8.22	18s	28.37/8.00	11s	28.42/8.04	14s
27.57/7.81	19s	27.09/7.63	11s	27.11/7.64	13s
25.82/7.23	21s	25.37/6.98	11s	25.56/7.12	13s
28.12/7.89	29s	27.53/7.74	13s	27.58/7.76	13s
25.23/6.70	26s	24.78/6.55	13s	24.80/6.56	13s
36.61/9.33	56s	36.02/9.28	28s	36.04/9.28	24s

Extended Data Fig. 8 | Performance of Noise2Fast with an hourglass architecture. Performance of Noise2Fast using the same architecture as DIP architecture, both validated using the original noisy image and validated using ground truth image.

Method	Noise2Fast	Noise2Self + Noise2Fast architecture	Noise2Self	Self2Self + Noise2Fast architecture	Self2Self
Time until PSNR >27.3	4s	42s	62s	612s	468s



Extended Data Fig. 9 | Performance of other methods using Noise2Fast Architecture. Performance over time of Self2Self and Noise2Self on the parrot image from Set12 with $\sigma = 25$ Gaussian noise, using both their standard architectures and the Noise2Fast architecture.

Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see our [Editorial Policies](#) and the [Editorial Policy Checklist](#).

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a Confirmed

- | | | |
|-------------------------------------|-------------------------------------|--|
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | The exact sample size (n) for each experimental group/condition, given as a discrete number and unit of measurement |
| <input type="checkbox"/> | <input checked="" type="checkbox"/> | A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | The statistical test(s) used AND whether they are one- or two-sided
<i>Only common tests should be described solely by name; describe more complex techniques in the Methods section.</i> |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | A description of all covariates tested |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals) |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | For null hypothesis testing, the test statistic (e.g. F , t , r) with confidence intervals, effect sizes, degrees of freedom and P value noted
<i>Give P values as exact values whenever suitable.</i> |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes |
| <input checked="" type="checkbox"/> | <input type="checkbox"/> | Estimates of effect sizes (e.g. Cohen's d , Pearson's r), indicating how they were calculated |

Our web collection on [statistics for biologists](#) contains articles on many of the points above.

Software and code

Policy information about [availability of computer code](#)

Data collection	To acquire images we used NIS elements (https://www.microscope.healthcare.nikon.com/products/software/nis-elements). To analyze, crop and LUT images we used ImageJ (https://imagej.nih.gov/ij/).
Data analysis	Our own denoiser along with all the methods to which we compared ourselves, our script for calculating PSNR/SSIM, and our script for adding synthetic Gaussian noise, and all benchmarking datasets are included on GitHub: https://github.com/pelletierlab/noise2fast . We also use CellPose for segmentation which is available here: https://github.com/MouseLand/cellpose .

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research [guidelines for submitting code & software](#) for further information.

Data

Policy information about [availability of data](#)

All manuscripts must include a [data availability statement](#). This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

We have made use of BSD68 dataset, available here: <https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/>, Set12 which is available here: <https://paperswithcode.com/dataset/set12> and the Fluorescence Microscopy Dataset (FMD) available here: <https://curate.nd.edu/show/f4752f78z6t>. Benchmarking datasets along with code and reproducibility instructions for Figure 2 are available on our GitHub (<https://github.com/pelletierlab/Noise2Fast>). Note that all speed benchmarks were performed on an RTX 5000 mobile GPU, and therefore results may vary according to GPU used. Source input and output images used to make the graphs in Figure 4 are publicly available on our GitHub (Noise2Fast/Fig5Data). Source data for Figure 3 is available on our GitHub as well (Noise2Fast/livecells).

Figure 1 is a conceptual illustration and does not make use of any datasets, however the image we use to illustrate checkerboard downsampling is a crop of an image available on our GitHub (Noise2Fast/BSD68/19.tif). The minimum dataset for the experiment illustrated in Figure 5 is publicly available on our GitHub (Noise2Fast/Fig6Data).

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

Life sciences Behavioural & social sciences Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see [nature.com/documents/nr-reporting-summary-flat.pdf](https://www.nature.com/documents/nr-reporting-summary-flat.pdf)

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	Sample sizes were selected to maximize comparability to other methods. For example, the BSD68 dataset contains 68 images and is widely used for benchmarking Gaussian denoisers, the same is true of Set12. For FMD we selected 5 diverse images because of the massive computational resources required to run some of our compared methods to completion on a 512x512 image.
Data exclusions	For FMD we selected 5 representative images to use as benchmark due to computational limitations, however this selection was done at the outset before any testing was done and our results on this dataset agree with our results on other datasets, at least in terms of how each method ranks.
Replication	We have made our code publicly available with full reproducibility instructions. The time, technical skill, and computational resources required to reproduce our results are minimal.
Randomization	Synthetic Gaussian noise was randomly generated using pythons build in random number generator.
Blinding	Blinding is not applicable to this study.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Included in the study
<input checked="" type="checkbox"/>	<input type="checkbox"/> Antibodies
<input type="checkbox"/>	<input checked="" type="checkbox"/> Eukaryotic cell lines
<input checked="" type="checkbox"/>	<input type="checkbox"/> Palaeontology and archaeology
<input checked="" type="checkbox"/>	<input type="checkbox"/> Animals and other organisms
<input checked="" type="checkbox"/>	<input type="checkbox"/> Human research participants
<input checked="" type="checkbox"/>	<input type="checkbox"/> Clinical data
<input checked="" type="checkbox"/>	<input type="checkbox"/> Dual use research of concern

Methods

n/a	Included in the study
<input checked="" type="checkbox"/>	<input type="checkbox"/> ChIP-seq
<input checked="" type="checkbox"/>	<input type="checkbox"/> Flow cytometry
<input checked="" type="checkbox"/>	<input type="checkbox"/> MRI-based neuroimaging

Eukaryotic cell lines

Policy information about [cell lines](#)

Cell line source(s)	MDA-MB 231 cell line was provided by Dr. Robert S. Kerbel (Sunnybrook Health Sciences Centre, Toronto, Canada) and RPE-1 cell line was purchased from American Type Culture Collection (ATCC)
Authentication	None of the cell lines were authenticated
Mycoplasma contamination	MDA-MB 231 and RPE-1 cell lines were negative for mycoplasma
Commonly misidentified lines (See ICLAC register)	<i>Name any commonly misidentified cell lines used in the study and provide a rationale for their use.</i>