Supplemental Information for "Understanding the Influence of Receptive Field and Network Complexity in Neural-Network-Guided TEM Image Analysis"

Network Architectures and their Receptive Fields

N=2		*N=3*		*N=4*	
Max Pooling kernel sizes (pixels)	Receptive Field (pixels)	Max Pooling kernel sizes (pixels)	Receptive Field (pixels)	Max Pooling kernel sizes (pixels)	Receptive Field (pixels)
[2,2]	44	[2,2,2]	96	[2,2,2,2]	200
[2,4]	64	[2,2,4]	136	[2,2,2,4]	280
[4,2]	80	[2,4,2]	168	[2,2,4,2]	344
[2,8]	104	[4,2,2]	184	[2,4,2,2]	376
[4,4]	120	[2,2,8]	192	[4,2,2,2]	392
[8,2]	152	[2,4,4]	248	[2,2,2,8]	440
[4,8]	200	[4,2,4]	264	[2,2,4,4]	504
[8,4]	232	[2,8,2]	312		
[8,8]	392	[4,4,2]	328		
		[8,2,2]	360		
		[4,4,4]	488		

Table S1: Network architectures and their calculated receptive fields when using max pooling to increase receptive field. N refers to the number of residual blocks in the UNet encoding (and decoding) arm. The max pooling kernel sizes are denoted as $[k_1, k_2, \ldots]$ where k_i is the kernel size of the max pooling layer after the ith residual block. Since the UNet is symmetric, the corresponding upsampling layers share the same kernel size. The receptive field can be converted into nanometers using the pixel size given in Table 1.

N=2		*N=3*		*N=4*	
Dilation Parameters	Receptive Field (pixels)	Dilation Parameters	Receptive Field (pixels)	Dilation Parameters	Receptive Field (pixels)
[1,1,1]	44	[1,1,1,1]	96	[1,1,1,1,1]	200
[1,1,2]	60	[1,1,1,2]	128	[1,1,1,1,2]	264
[1,2,2]	76	[1,1,2,2]	160	[1,1,1,2,2]	328
[2,2,2]	84	[1,2,2,2]	176	[1,1,2,2,2]	360
[1,2,3]	92	[1,1,2,3]	192	[1,2,2,2,2]	376
[1,3,3]	108	[1,2,2,3]	208	[1,1,1,2,3]	392
[2,3,3]	116	[1,2,3,3]	240	[1,1,2,2,3]	424
[3,3,3]	124				

Table S2: Network architectures and their calculated receptive fields when using dilated convolution to increase receptive field. N refers to the number of residual blocks in the UNet encoding (and decoding) arm. The dilation parameters are denoted as $[\alpha_1, \alpha_2, \ldots]$ where α_i is the dilation parameter of the convolution layers in the ith residual block in the encoding arm. Since the UNet is symmetric, the corresponding decoding residual block shares the same dilation parameter. Note that there are $N+1$ dilation parameters listed because there is a residual block between the encoding and decoding arms in the UNet. The receptive field can be converted into nanometers using the pixel size given in Table 1.

Results with Early Stopping

To prove that our results are not influenced by the choice of a constant number of training epochs, we implemented early stopping and have replicated our results in Figure 2d. In our implementation of early stopping, we save the model with the lowest validation loss until the early stopping criteria is satisfied. Our stopping criteria is that four consecutive validation losses are higher than the average validation loss of their last eight epochs.

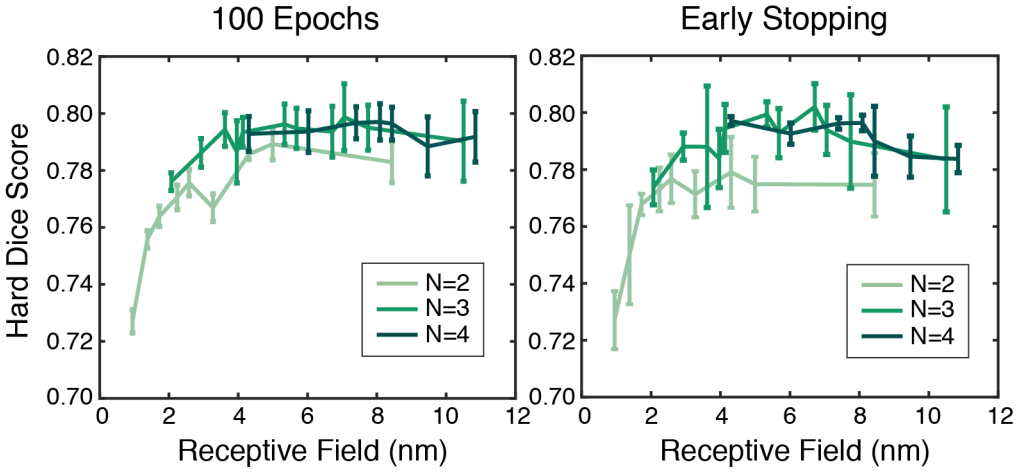

Figure S1: Comparing early stopping against a constant number of training epochs with the high-resolution dataset from Figure 2d. On the left is a replotting of Figure 2d and on the right is our results with early stopping. N refers to the number of residual blocks in the network (and therefore its complexity).

As seen in Figure S1, the choice in number of training epochs does not change our conclusions. We compare our results from Figure 2d (100 epochs of training) with the results from early stopping, in which the initial conditions are all the same such that the only difference is number of training epochs. While the exact numbers between the two methods differ slightly, the two graphs show the same behavior: an increase in performance with larger receptive field, and a smaller increase in performance as the networks become more complex. Early stopping leads to larger variations in the standard deviation. As our validation loss curve is not perfectly smooth, there is greater variation in which epoch would trigger early stopping.

Soft Dice Score Results

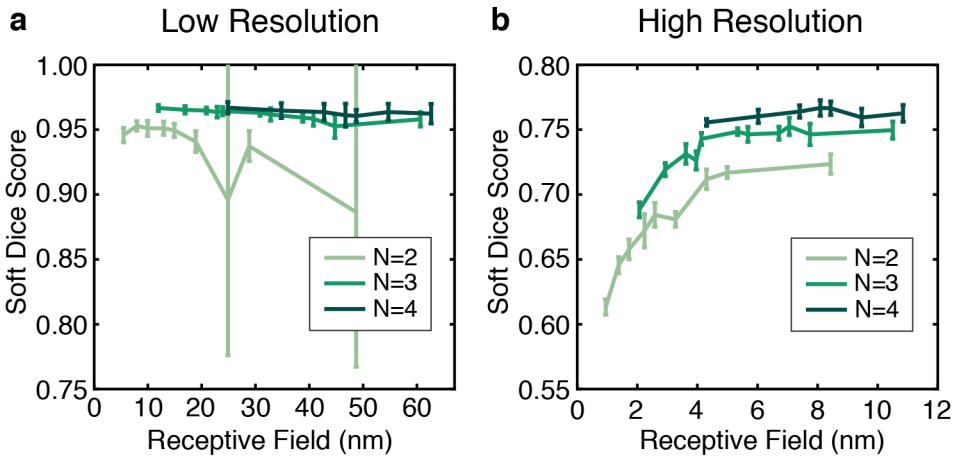

Figure S2: Soft dice scores for Figure 2, or low resolution and high resolution TEM images. (a) Soft dice score for Figure 2c. (b) Soft dice score for Figure 2d.

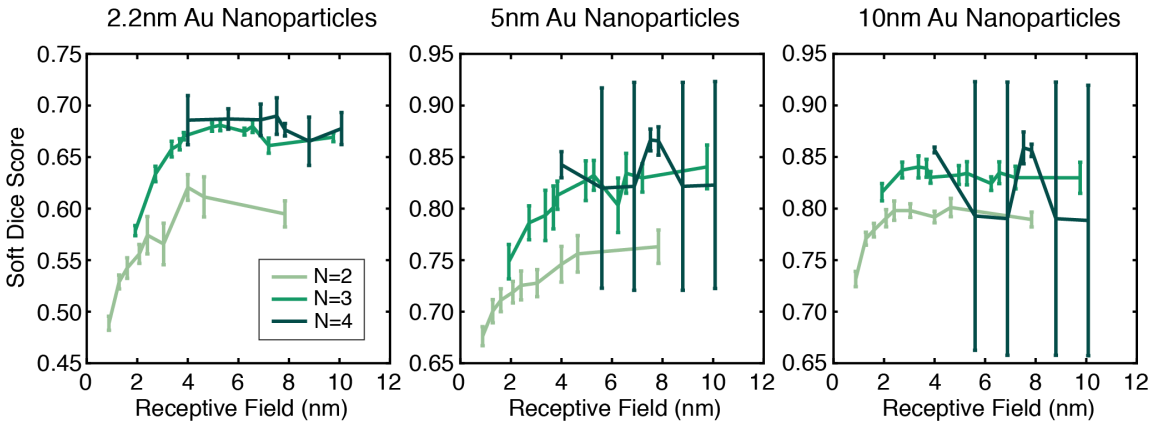

Figure S3: Soft dice scores for Figure 3, or high resolution TEM images of nanoparticles of various diameters.

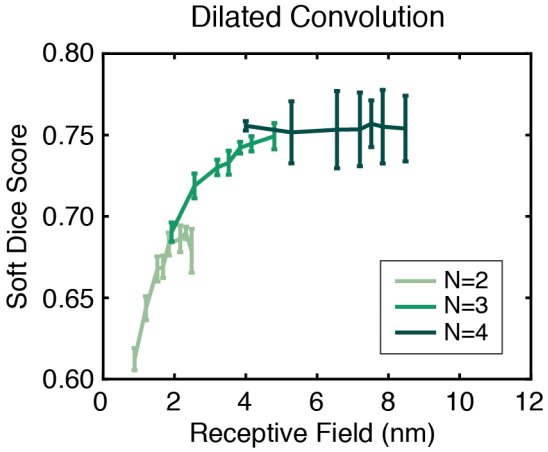

Figure S4: Soft dice scores for Figure 4b, or by increasing receptive field using dilated convolution.

Here we provide the soft dice scores for the receptive field studies shown in Figures 2, 3, and 4. The soft dice score differs from the hard dice score by using the output pixel's class probability rather than its binary prediction to calculate the dice score. Therefore, the soft dice score will give a better idea as to the network's confidence in its predictions. For example, as seen in Figures S2a and S3, there are a number of architectures with large error bars because one of the runs had a very low soft dice score. However, we do not see that same behavior in the hard dice score prediction, showing that a network can technically perform well even if it is not sure in its prediction.

From these soft dice score plots, it is also more clear as to how complexity helps a neural network. For example, in Figure S2b, it is clearer that more complex networks help increase confidence. In the corresponding hard dice score plot in Figure 2d, networks with the same receptive field but different complexities all perform similarly. However, when we look at the soft dice score, we see that for the same receptive field, more complex networks are more confident in their prediction.

Interpreting the Dice Score

As mentioned in the main text, the dice score penalizes undersegmentation (false negatives) more than oversegmentation (false positives). We can mathematically demonstrate this with the following proof:

For binary classification, the dice score can be rewritten as:

$$D = \frac{2TP}{2TP + FP + FN} \tag{1}$$

where TP refers to the number of true positive pixels, FP the number of false positive pixels, and FN the number of true negative pixels. Let's define a as the number of pixels in the ground truth label that is labeled as nanoparticle, b the number of pixels in the ground truth label that is labeled as background, and c the number of pixels that the network output has labeled wrong. Therefore, $a + b$ is the total number of pixels in the image.

If the network outputs an oversegmented image where c pixels that are actually background are mislabeled as nanoparticle, then the number of true positives is $TP = a$, the number of true negatives is $TN = b - c$, the number of false positives is $FP = c$, and the number of false negatives is $FN = 0$. Therefore, the dice score of this oversegmented image is

$$D_{\text{over}} = \frac{2a}{2a + c} \tag{2}$$

If the network instead outputs an undersegmented image where c pixels that are actually nanoparticle are mislabeled as background, then the number of true positives is now $TP = a - c$, the number of true negatives is $TN = b$, the number of false positives is $FP = 0$, and the number of false negatives is $FN = c$. The dice score of this undersegmented image is

$$D_{\text{under}} = \frac{2(a-c)}{2(a-c) + c} = \frac{2(a-c)}{2a - c} \tag{3}$$

After some algebra, we can show that D_{under} is always smaller than D_{over}

$$\begin{aligned} D_{\text{over}} &\overset{?}{\geq} D_{\text{under}} \\ \frac{2a}{2a+c} &\overset{?}{\geq} \frac{2(a-c)}{2a-c} \\ 4a^2 - 2ac &\overset{?}{\geq} 4a^2 - 2ac - 2c^2 \\ 0 &\geq -2c^2 \end{aligned}$$

Therefore, for a constant number of "wrong" pixels, the dice score penalizes undersegmentation more than oversegmentation.

Additional segmentation results on high-resolution TEM dataset

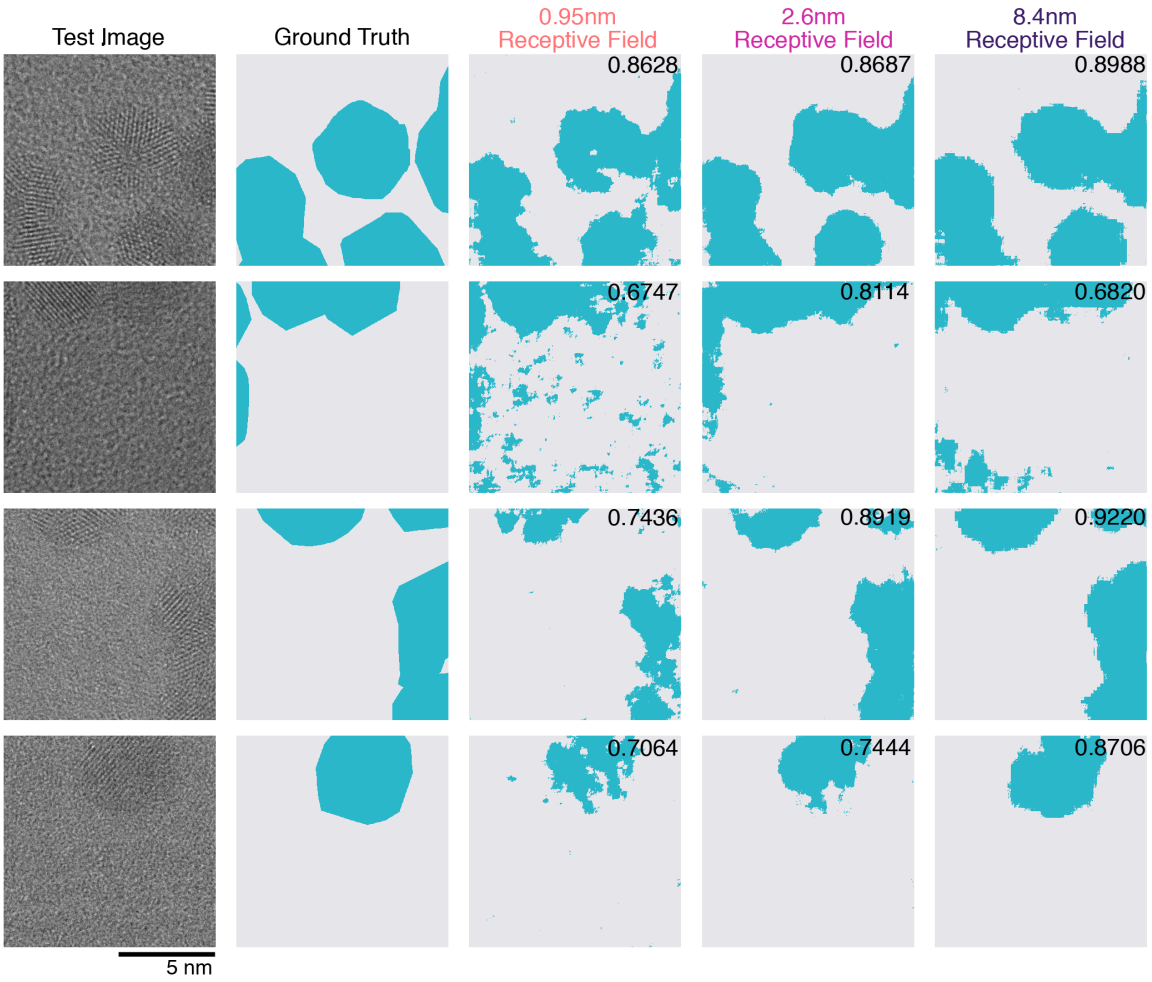

Figure S5: Additional segmentation results on the high resolution dataset. The three chosen networks are the same as in Figure 6 in the main text: $N = 2$ networks which only vary in receptive field. The hard dice score of each prediction is displayed in the upper right hand corner.

Examples of high-resolution 2.2nm nanoparticle segmentation results

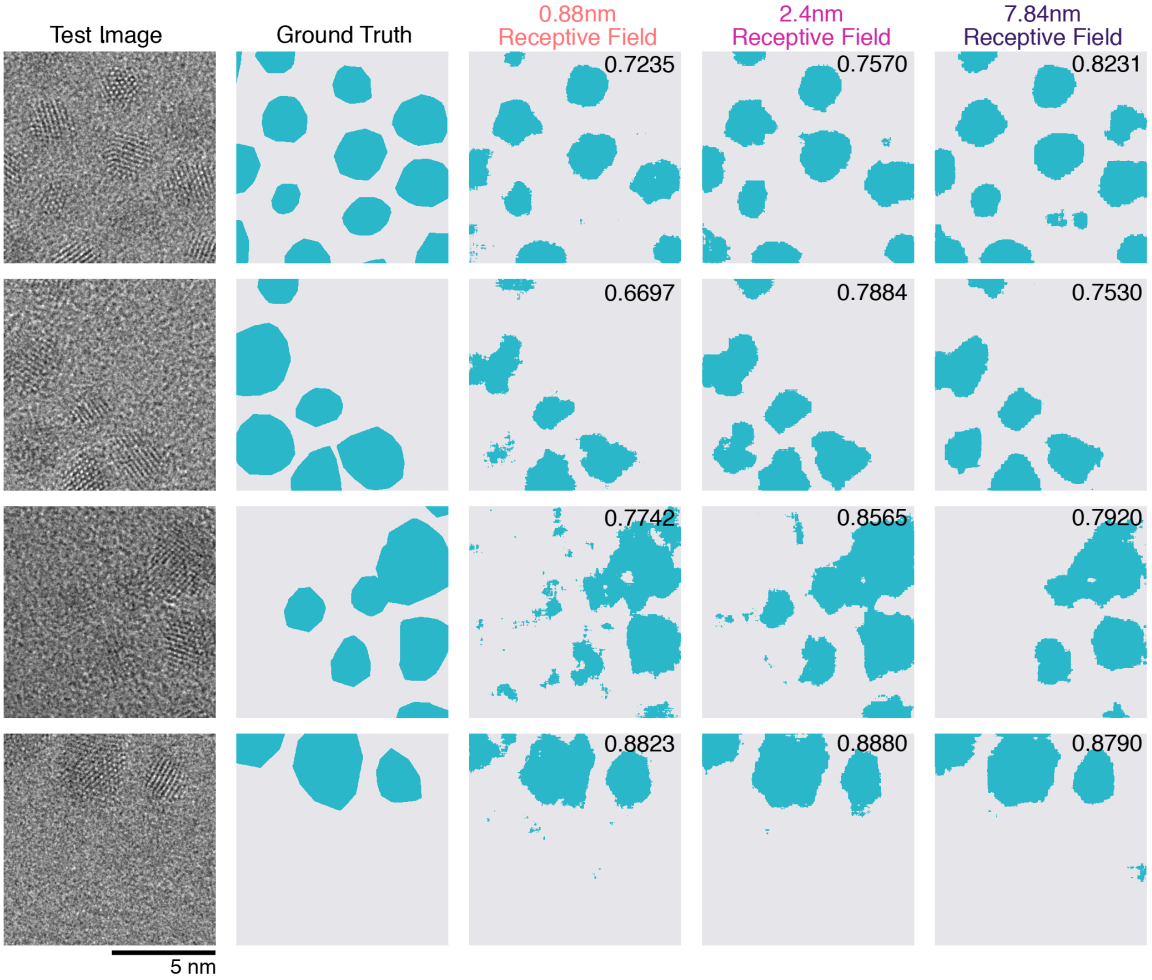

Figure S6: Example segmentation results on the 2.2nm dataset, used in Figure 3. The three chosen networks are 2-residual-block networks with varying receptive field. The hard dice score of each prediction is displayed in the upper right hand corner.

Examples of high-resolution 10nm nanoparticle segmentation results

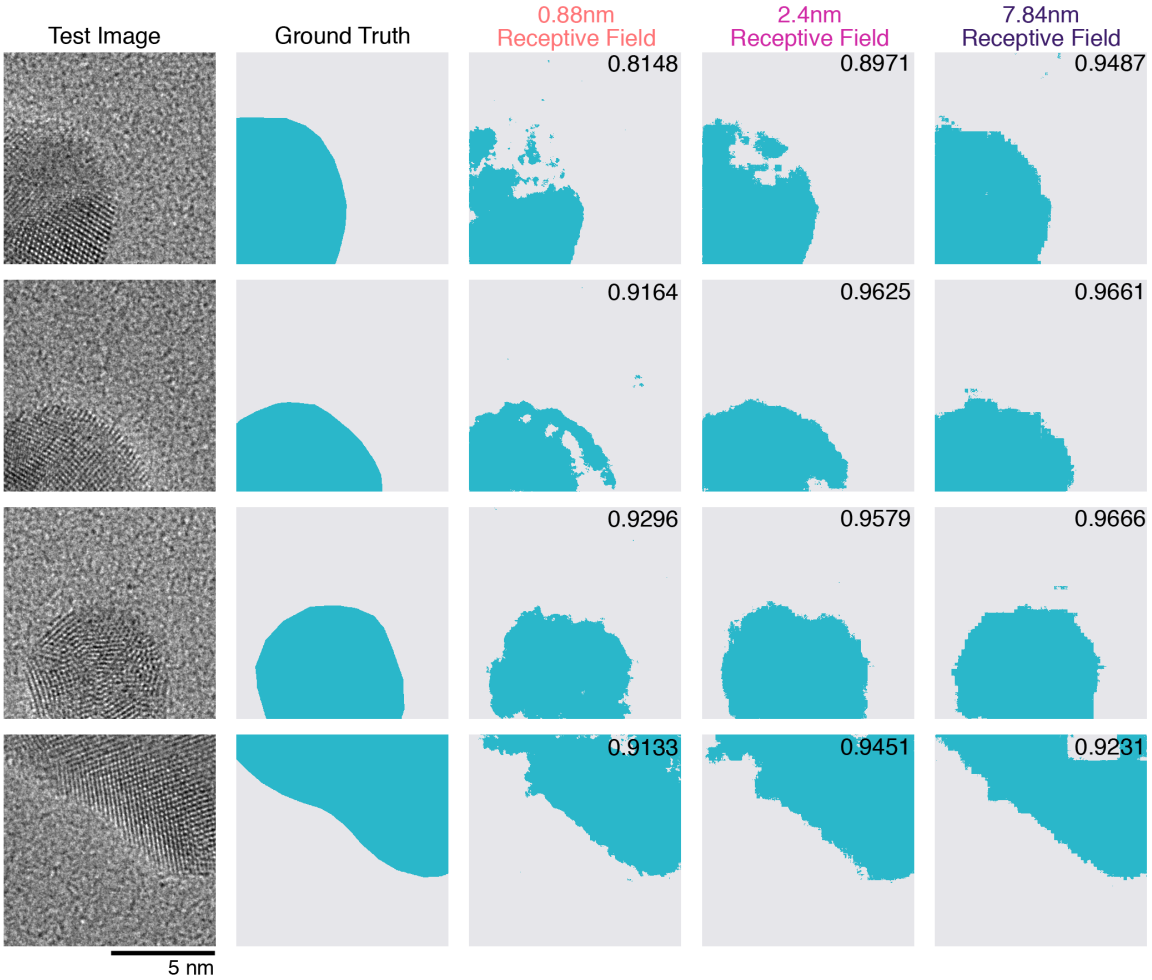

Figure S7: Example segmentation results on the 10nm dataset, used in Figure 3. The three chosen networks are 2-residual-block networks with varying receptive field. The hard dice score of each prediction is displayed in the upper right hand corner.

Additional Fourier Filtering Comparison Examples

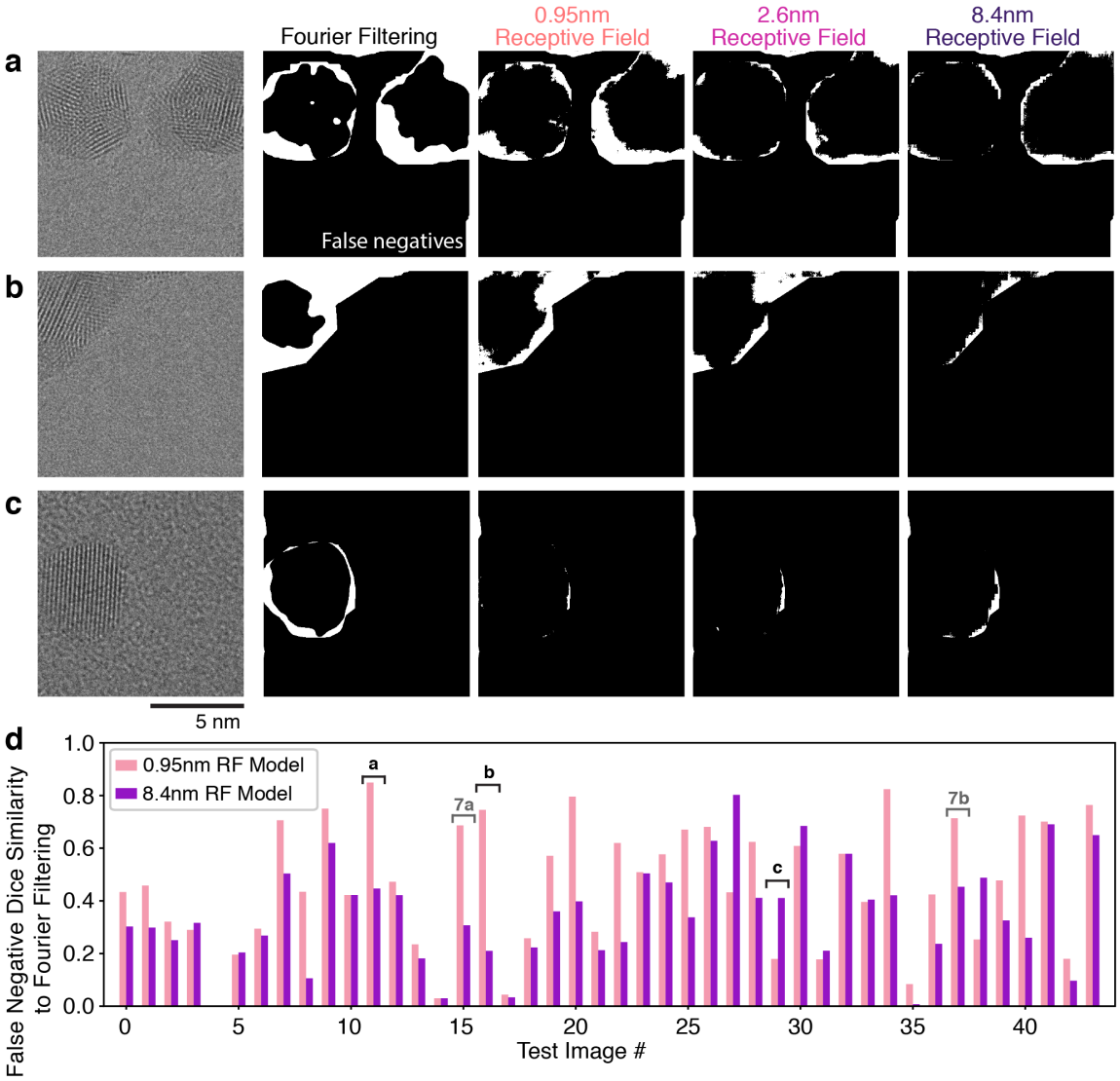

Figure S8: Additional comparisons between the false negative regions using Fourier filtering versus neural networks with varying receptive fields. (a,b,c) Examples from the test set and their corresponding false negative regions after either Fourier filtereing or using a 2-residual-block neural network with either 0.95nm, 2.6nm, or 8.4nm receptive field. The same neural networks were chosen as Figure 7. (d) Dice scores of the false negative regions of the 0.95nm receptive field and the 8.4nm receptive field networks in comparison to the false negative regions from Fourier filtering for all test images. Labels denote the dice similarity scores for the three examples in (a), (b), and (c), as well as the two examples in Figure 7.

To quantify the similarity between results from Fourier filtering and the small receptive field network, we calculate the dice score between the false negative regions from the 0.95nm receptive field model and the false negative regions from Fourier filtering for all 44 non-augmented test images. For comparison, we also calculate the dice score similarity between the false negative regions from Fourier filtering and a 8.4nm receptive field network. The similarity scores for all test images are shown in Figure S8d, with the examples shown in Figure 6 and Figure S8 highlighted. We see that for most images, the false negative dice score similarity is greater between the small receptive field and Fourier filtering than between the large receptive field and Fourier filtering. We note that this false negative dice score is not a direct measure of segmentation performance as it utilizes the Fourier filtering false negatives as a ground truth.

Fourier filtering on the 2.2nm, 5nm, and 10nm datasets

Dataset	2.2nm Au	5nm Au	10nm Au
Fourier Filtering Average Dice Score	0.5056	0.6105	0.6208

Table S3: Average dice score of the segmentation results from Fourier filtering on the 2.2nm, 5nm, and 10nm high-resolution TEM datasets.

In Table S3, we calculate the average dice score using Fourier filtering for the 2.2nm, 5nm, and 10nm diameter Au nanoparticle datasets. The higher Fourier filtering dice score for the 10nm Au dataset suggests that more of its nanoparticle regions have visible lattice fringes.

Effective Receptive Field

The effective receptive field is the weighted area of the input image that contributes to the output decision. In practice, this can be computed as:

$$ERF(i,j) = \frac{\partial y}{\partial x_{ij}} \quad \forall x_{ij} \in X$$

where y is the decision pixel and x_{ij} is the pixel at the (i,j) location in the input image X (Luo, et al. 2016). To visualize and quantify the effective receptive field, we input 32 randomly initialized images (mean of 0, standard deviation of 1, similar to the input TEM images) to our already trained neural networks for the 2.2nm Au dataset (the neural networks used in Figure 3) and calculate the above partial derivative for the center pixel in the output image using backpropagation. We then take the average over all (32 effective receptive fields) $\times$ (5 runs) = 160 images and normalize to get the effective receptive field for each architecture.

In Figure S9, we show the effective receptive field for different architectures of the 2-residual-block UNet with varying (maximal) receptive fields. In Figure S9a, we see that not all pixels within the theoretical receptive field contribute equally to the output decision, and most of the influence lies with the pixels closest to the center (decision) pixel. As the maximal receptive field increases (denoted by the red box), most of the influence stays around the center pixel, as highlighted by the zoom-in in Figure S9b. However, the outlying pixels do contribute, as evidenced by the line-cuts in Figure S9c which shows non-zero contributions from the outer pixels and zero contribution from pixels outside the maximal receptive field. Our effective receptive fields do not show a Gaussian-like shape, as found in Luo, et al (2016), because our network architecture has skip connections which allow input pixels close to the output decision pixel to more directly influence the decision.

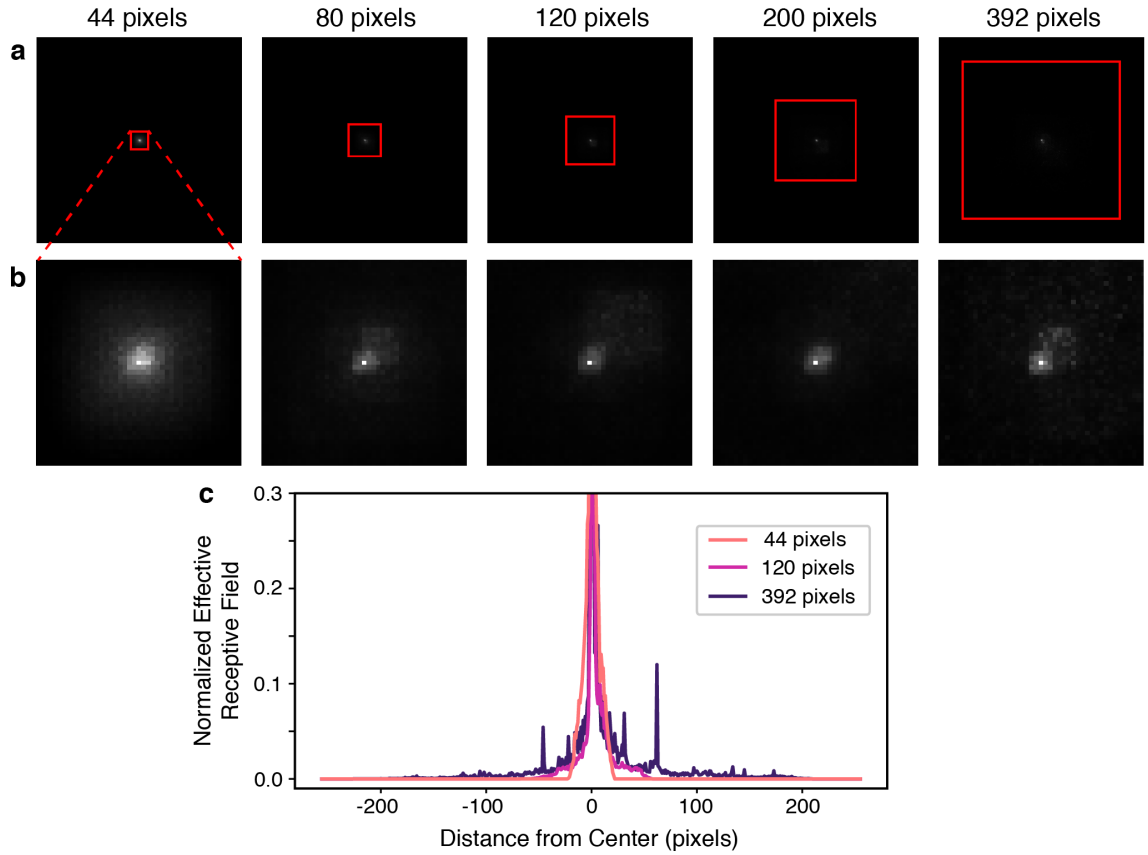

Figure S9: Effective receptive field of the 2-residual-block UNet for various maximal receptive fields. (a) The effective receptive field of a trained 2-residual-block UNet on the 2.2nm Au dataset for various maximal receptive fields (see Table S1 for more details). Red box shows the theoretical maximal receptive field size that is reported in the main text. (b) Zoom in of the 44x44 pixel region around the center of the effective receptive field for the various maximal receptive fields. (c) Line cut through the center of the normalized effective receptive field for the 44 pixel, 120 pixel, and 392 pixel maximal receptive field networks.