Research Article Vol. 4, No. 9 / September 2017 / Optica 1117

Lensless computational imaging through deep learning

Ayan Sinha,[1,*] Justin Lee,[2] Shuai Li,[1] and George Barbastathis[1,3]

[1]Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, Massachusetts 02139, USA
[2]Institute for Medical Engineering Science, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, Massachusetts 02139, USA
[3]Singapore-MIT Alliance for Research and Technology (SMART) Centre, One Create Way, Singapore 117543, Singapore
*Corresponding author: sinhayan@gmail.com

Received 27 June 2017; revised 9 August 2017; accepted 10 August 2017 (Doc. ID 300937); published 15 September 2017

Deep learning has been proven to yield reliably generalizable solutions to numerous classification and decision tasks. Here, we demonstrate for the first time to our knowledge that deep neural networks (DNNs) can be trained to solve end-to-end inverse problems in computational imaging. We experimentally built and tested a lensless imaging system where a DNN was trained to recover phase objects given their propagated intensity diffraction patterns.

OCIS codes: (100.3190) Inverse problems; (100.4996) Pattern recognition, neural networks; (100.5070) Phase retrieval; (110.1758) Computational imaging.

https://doi.org/10.1364/OPTICA.4.001117

1. INTRODUCTION

Neural network training can be thought of as a generic function approximation: given a training set (i.e., examples of matched input and output data obtained from a hitherto-unknown model), a neural network attempts to generate a computational architecture that accurately maps all inputs in a test set (distinct from the training set) to their corresponding outputs. The basic idea dates back to the 1950s [1]; recently, Deep Neural Networks (DNNs), so called because they are structured as multilayered with a large number of layers, have found extensive use. To generalize well, DNNs exploit several innovations in connectivity, for example, convolutions [2–5] for regularization and pruning in image recognition and classification tasks; nonlinearities, such as non-differentiable piecewise linear units [6] as opposed to the older sigmoidal functions that were differentiable but also prone to stagnation [7]; and algorithms, such as more efficient backpropagation (backprop) [8,9]. In turn, these have been applied in diverse domains: playing complex games on Atari 2600 [10] and Go [11]; object generation [12]; object detection [13]; and image processing: colorization [14], deblurring [15–17], and in-painting [18]. In this paper, we propose that DNNs may "learn" to approximate solutions to inverse problems in computational imaging.

A general computational imaging system consists of a physical part and a computational part. In the physical part, light propagates through one or more objects of interest as well as optical elements such as lenses, prisms, etc., finally producing a raw intensity image on a digital camera. The raw intensity image is then computationally processed to yield object attributes, e.g., a spatial map of light attenuation and/or phase delay through the object—what we traditionally call "image" and "quantitative phase image," respectively. The computational part of the system is then said to have produced a solution to the inverse problem.

The specific computational imaging scenario chosen for this paper is quantitative phase retrieval. Let $\psi(x,y;z=0)=\exp[i\phi(x,y)]$ represent the optical field purely phase-modulated by the unknown object $\phi(x,y)$ as function of the lateral Cartesian coordinates x, y at the origin $z=0$ of the optical axis. The measurement after propagation by distance z is the raw intensity image $I(x,y)=|\psi(x,y;z)|^2\equiv H\phi(x,y)$, where H denotes the forward operator relating the phase $\phi(x,y)$ at the origin to the raw intensity image $I(x,y)$ at distance z. The problem of phase retrieval, then, is to formulate an inverse transform H^{inv}, generally nonlinear, such that

$$\hat{\phi}(x,y)=H^{\text{inv}}I(x,y) \tag{1}$$

represents an acceptable estimate of the phase object. The Gerchberg–Saxton–Fienup (GSF) class of algorithms [19,20] is a well-known and highly successful approach where instead of an explicit H^{inv}, the inverse $\hat{\phi}(x,y)$ is estimated iteratively from a single raw intensity image. Two other notable approaches are: digital holography (DH) [21], where an additional reference beam $\exp[ikx]$ is assumed to be present so that instead an interferometric measurement $I_{\text{holo}}=|\exp[ikx]+\psi(x,y;z)|^2$ is available; and transport of intensity (TIE) [22,23], utilizing an "axial stack" of one or more additional raw intensity images $I_1(x,y)=|\psi(x,y;z+\delta z_1)|^2$, $I_2(x,y)=|\psi(x,y;z+\delta z_2)|^2$, …, where $\delta z_1, \delta z_2, \ldots$ are axial displacements. On-axis DH ($k=0$) requires the object to be spatially sparse so that the incoming light remains mostly unscattered because it must serve as reference

beam for the digital hologram; off-axis DH ($k \neq 0$) requires the object to be low-bandwidth so that the space-bandwidth product limitation of the camera measuring I_{holo} is not violated. On the other hand, TIE requires sparse object gradients to avoid singularities as result of propagation.

Another approach to obtain H^{inv} is by solving an optimization problem of the form

$$\hat{\phi}(x,y) = \text{argmin}_{\phi} \|H\phi - I\|^2 + \alpha\Phi(\phi). \quad \textbf{(2)}$$

Here, H is the same forward operator, I is the measurement (including noise), Φ is the regularizer expressing prior information about the object, and α is the regularization parameter that controls the relative strength of the two terms in the optimization functional. This computational approach, central to many inverse problems [24], is traced back to Tikhonov [25] and Wiener [26]. For Eq. (2) to work well, in addition to the obvious concerns of convexity and convergence speed, H and Φ need to be precisely known or at least discoverable. For Φ, expressions promoting sparsity have been often proven to be effective [27–29]; on the other hand, the damage due to incomplete or erroneous knowledge in H, for example, due to insufficient accuracy in the approximations, or due to experimental errors such as misalignments, is more difficult to control [30].

The hypothesis that we set out to test in this paper is whether a DNN can be trained to act as the inverse operator H^{inv} in Eq. (1). In addition to the well-established adaptability of DNNs as effective approximators to general-purpose nonlinear operators, our approach is attractive for several reasons: the DNN, once trained, should recover every possible object within a hopefully large enough class, unlike GSF where H^{inv} is obtained separately and iteratively for each new object presented. Alternatively, since during DNN training iteration takes place over several objects simultaneously, the learning approach is expected to be a more robust generalization of GSF. Moreover, the DNN approach is not subject to any of the sparsity requirements of either DH or TIE. Indeed, our experiment, described in detail in Section 2, was explicitly designed to violate the DH and TIE assumptions. Lastly, compared to optimization-based approaches, the DNN input–output relationship (which is in general too complicated to be written in explicit form) replaces the form in Eq. (2). The user is therefore not required to specify the forward operator H and the prior Φ or decide about the magnitude of the regularization parameter α; instead, to the degree that our results show successful inversion, they suggest that the DNN learns H, Φ, and α implicitly through training. The downside of the DNN approach is that a sufficiently large database of known pairs of phase objects and their corresponding raw intensity images must be available for training and testing; we will return to this issue shortly.

Neural networks have been used in a variety of imaging and image restoration tasks. For example, Hopfield's associative memory network [31] was capable of retrieving entire faces from partially obscured inputs, and was implemented in an all-optical architecture [32]. Recently, Horisaki *et al.* [33] used support-vector machines, an older form of bilayer neural network with nonlinear discriminant functions, also to recover face images through a scattering medium; the specific scatterer was learned by the neural network in lieu of the more common approach of having the scatterer characterized through a transmission matrix [34,35]. Modern deep networks and training schemes have been used to model (learn) the scatterer's forward operator using an architecture designed to emulate the layered structure of the scatterer itself [36]. More recently, DNNs have also been used for classification of cells from raw intensity images captured in a holographic arrangement [37] and improvement in image quality by first forming images with traditional means, and then using a DNN trained from object-image pairs to sharpen the visual image quality. DNN-enabled image restoration has been carried in a variety of forward operators: tomography [38], digital holography [39], and standard microscopy [40]. Our approach is, instead, truly end-to-end: the DNN implements the H^{inv} operator from Eq. (1) in its entirety; the first such end-to-end attempt, to our knowledge, is an earlier, more primitive effort by our own group with binary phase objects [41]. Moreover, because the phase objects in both our earlier and current efforts are generated by a precisely calibrated spatial light modulator, we are able to quantitatively evaluate the fidelity of our DNN-generated estimates $\hat{\phi}(x,y)$ to the true objects $\phi(x,y)$.

Neural network approaches often come under criticism because the quality of training depends on the quality of the examples given to the network during the training phase. For instance, if the inputs used to train a network are not diverse enough, then the DNN will learn priors of the input images instead of generalized rules for phase retrieval from intensity. This was the case in Ref. [33], where a support vector machine (SVM) trained using images of faces could adequately reconstruct faces, but when given the task of reconstructing images of natural objects such as a pair of scissors, the trained SVM still returned an output that resembled a human face.

For our specific problem, an ideal training set would encompass all possible phase objects. Unfortunately, phase objects, generally speaking, constitute a rather large class, and it would be unrealistic to attempt to train a network sampling from across all possible objects from this large class. Instead, we synthesize phase objects in the form of natural images derived from the ImageNet [42] database because it is readily available and widely used in the study of various machine learning problems. For comparison, we also trained a separate network using a narrower class of (facial) images from the Faces-LFW [43] database.

As expected, our network did well when presented with unknown phase objects in the form of faces or natural images that it had been trained to. Notably, the network also performed well when presented with objects outside of its training class—the DNN trained using images of faces was able to reconstruct images of natural objects, and the DNN trained using images of natural objects was able to reconstruct images of faces. Additionally, both DNNs were able to reconstruct completely distinct images, including handwritten digits, characters from different languages (Arabic, Mandarin, English), and images from a disjoint natural image dataset. Both trained networks yielded accurate results, even when the object-to-sensor distance(s) in the training set slightly differed from that of the testing set, suggesting that the networks are not merely pattern-matching but have instead actually learned a generalizable model approximating the inverse operator H^{inv}.

The details of our experiment, including the physical system and the computational training and testing results, are described in Section 2. The neural network itself is analyzed in Section 3, and concluding thoughts are in Section 4.

2. EXPERIMENT

Our experimental arrangement is shown in Fig. 1. Light from a He–Ne laser source (Thorlabs, HNL210L, 632.8 nm) first transmits through a spatial filter, which consists of a microscope objective (Newport, M-60X, 0.85NA) and a pinhole aperture ($D = 5$ μm), to remove spatial noise. After being collimated by the lens ($f = 150$ mm), the light is reflected by a mirror and then passes through a linear polarizer, followed by a beam splitter. A spatial light modulator (Holoeye, LC-R 720, reflective) is placed normally incident to the transmitted light and acts as a pixel-wise phase object. The SLM pixel size is 20×20 μm^2, and the number of pixels is 1280×768, out of which the central 512×512 portion only is used in the experiments. The SLM-modulated light is then reflected by the beam splitter and passes through a linear polarization analyzer, before being collected by a CMOS camera (Basler, A504k) placed at a distance z. The CMOS camera pixel size is 12×12 μm^2, and the number of pixels is 1280×1024, which is cropped to a 1024×1024 square used for processing. The total used linear camera size of ≈12.3 mm is slightly larger than the active SLM size of ≈10.2 mm to accommodate expansion of the propagating beam due to diffraction. Images recorded by the CMOS camera are then processed on an Intel i7 CPU, with neural network computations performed on a GTX1080 graphics card (NVIDIA).

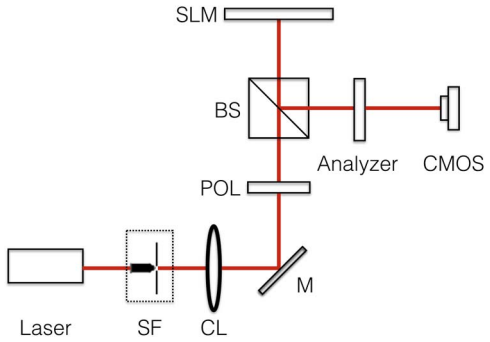

Fig. 1. Experimental arrangement. SF, spatial filter; CL, collimating lens; M, mirror; POL, linear polarizer; BS, beam splitter; SLM, spatial light modulator.

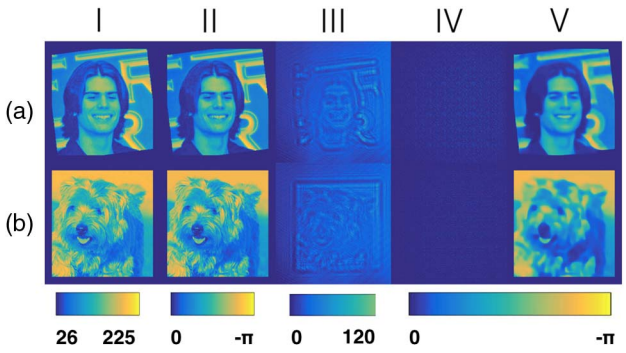

Fig. 2. DNN training. Rows (a) and (b) denote the networks trained on Faces-LFW and ImageNet datasets, respectively. (i) Randomly selected example drawn from the database; (ii) calibrated phase image of the drawn sample; (iii) diffraction pattern generated on the CMOS by the same sample; (iv) DNN output before training (i.e., with randomly initialized weights); (v) DNN output after training.

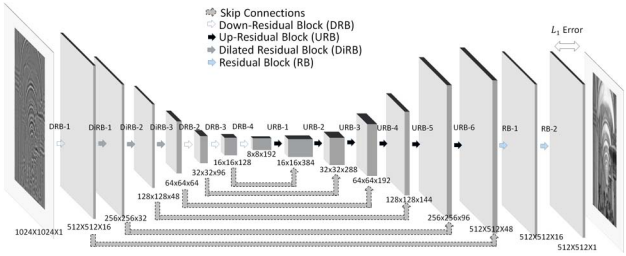

Fig. 3. Detailed schematic of our DNN architecture, indicating the number of layers, nodes in each layer, etc.

According to its user manual, the LC-R 720 SLM can realize (approximate) pure-phase modulation if we modulate the light polarization properly. Specifically, for He–Ne laser light, if we set the incident beam to be linearly polarized at 45° with respect to the vertical direction, and also set the linear polarization analyzer to be oriented at 340° with respect to the vertical direction, then the amplitude modulation of the SLM will become almost independent of the assigned (8-bit gray-level) input. In this arrangement, the phase modulation of the SLM follows a monotonic,

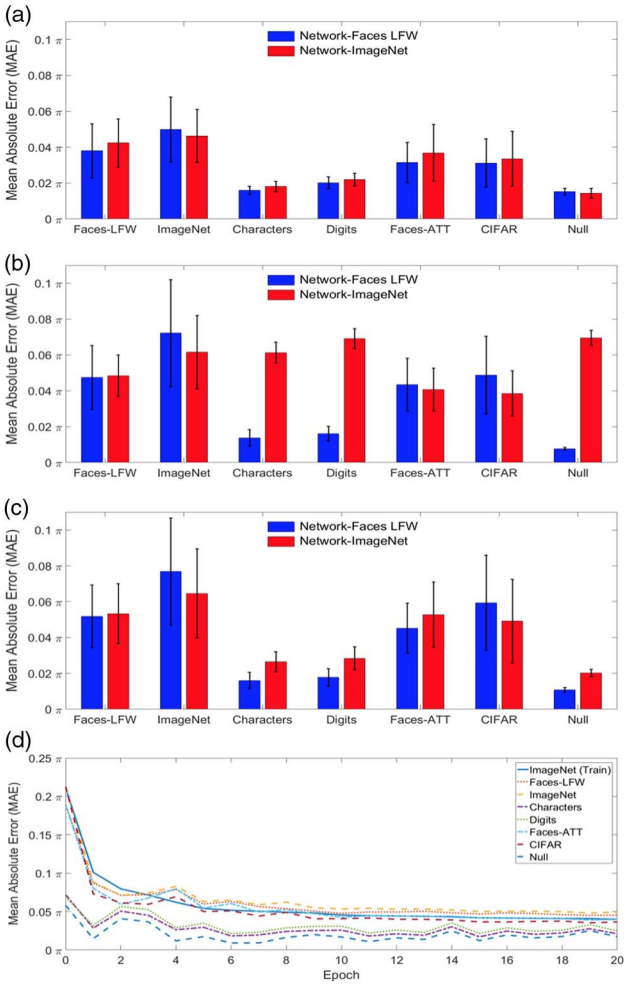

Fig. 4. Quantitative analysis of our trained deep neural networks for three object-to-sensor distances, (a) z_1, (b) z_2, and (c) z_3, for the DNNs trained on Faces-LFW (blue) and ImageNet (red) on seven datasets. (d) The training and testing error curves for the network trained on ImageNet at distance z_3 over 20 epochs.

almost-linear relationship with the assigned pixel value (with maximum phase depth: $\sim 1\pi$). We experimentally evaluated the correspondence between 8-bit grayscale input images projected onto the SLM and phase values in the range $[0, -\pi]$ (see Supplement 1). In this paper, we approximate our SLM as a pure-phase object and computationally recover the phase using a neural network.

The CMOS detector was placed after a free-space propagation distance z. Distinct DNNs were trained from recorded diffraction patterns at three distances $z_1 = 37.5\ \text{cm} \pm 2\ \text{mm}$ ($\text{NA} = 0.0164 \pm 9 \times 10^{-4}$, Fresnel number $\mathcal{F} = 159 \pm 1$), $z_2 = 67.5\ \text{cm} \pm 2\ \text{mm}$ ($\text{NA} = 0.0091 \pm 1 \times 10^{-4}$, Fresnel number $\mathcal{F} = 88.3 \pm 0.3$) and $z_3 = 97.5\ \text{cm} \pm 2\ \text{mm}$ ($\text{NA} = 0.0063 \pm 1 \times 10^{-4}$, Fresnel number $\mathcal{F} = 61.2 \pm 0.1$). Our experiment consisted of two phases: training and testing. During the training phase, we modulated the phase SLM according to samples randomly selected from the Faces-LFW or ImageNet database. We resized and padded selected images before displaying them on our SLM. Two examples of inputs, as they are sent to the SLM, and their corresponding raw intensity images (diffraction patterns), as captured on the CMOS, are shown in Fig. 2. Our training set consisted of 10,000 such faces/images—diffraction pattern pairs. The raw intensity images from all these training examples were used to train the weights in our DNN. We used a Zaber A-LST1000D stage with repeatability 2.5 μm to translate the camera in order to analyze the robustness of the learned network to axial perturbations. The positional accuracy of ± 2 mm reported earlier on the training distances z_1, z_2, z_3 is derived from our error in manually establishing the absolute distance between the SLM and camera, whereas the stage repeatability determines the error in camera displacement relative to these initial training positions, respectively, for each axial robustness test in Section 3.

Our DNN uses a convolutional residual neural network (ResNet) architecture. In a convolutional neural network (CNN), inputs are passed from nodes of each layer to the next, with adjacent layers connected by convolution. Convolutional ResNets extend CNNs by adding short-term memory to each layer of the network. The intuition behind ResNets is that one should only add a new layer if one stands to gain by adding that layer. ResNets ensure that the $(N+1)$th layer learns something new about the network by also providing the original input to the output of the $(N+1)$th layer and performing calculations on the residual of the two. This forces the new layer to learn something different from what the input has already encoded/learned [5].

A diagram of our specific DNN architecture is shown in Fig. 3. The input layer is the image captured by the CMOS camera. It is then successively decimated by seven residual blocks of convolution + downsampling followed by six residual blocks of deconvolution + upsampling, and finally two standard residual blocks. Some of the residual blocks comprise dilated convolutions so as to increase the receptive field of the convolution filters, and hence, aggregate diffraction effects over multiple scales [44]. We use skip connections to pass high-frequency information learned in the initial layers down the network toward the output reconstruction, and have two standard residual blocks at the end of the network to fine-tune the reconstruction. At the very last layer of our DNN, the values represent an estimate of our phase object. The connection weights are trained using backpropagation (not to be confused with optical backpropagation) on the L_1 distance between the network output and the nominal appearance of the training samples, represented as

$$\min \frac{1}{wh} \| Y - G \|_1. \tag{3}$$

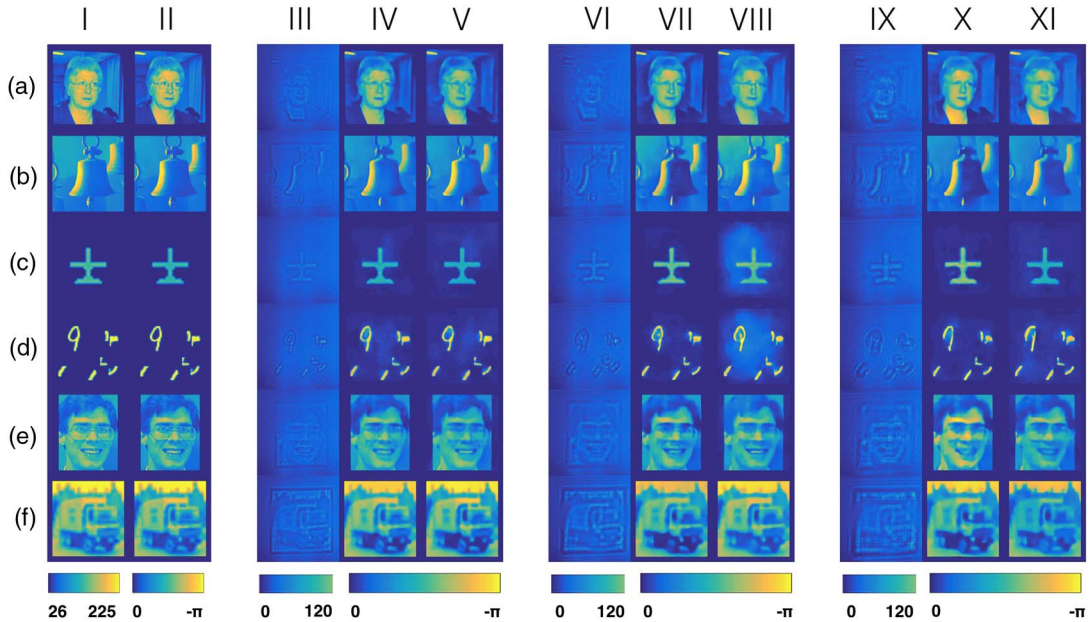

Fig. 5. Qualitative analysis of our trained deep neural networks for combinations of object-to-sensor distances z and training datasets. (i) Ground truth pixel value inputs to the SLM. (ii) Corresponding phase images calibrated by SLM response curve. (iii) Raw intensity images captured by CMOS detector at distance z_1. (iv) DNN reconstruction from raw images when trained using Faces-LFW [43] dataset. (v) DNN reconstruction when trained using ImageNet [42] dataset. Columns (vi–viii) and (ix–xi) follow the same sequence as (iii–v), but in these sets the CMOS is placed at a distance of z_2 and z_3, respectively. Rows (a)–(f) correspond to the dataset from which the test image is drawn: (a) Faces-LFW, (b) ImageNet, (c) Characters, (d) MNIST Digits, (e) Faces-ATT [46], or (f) CIFAR [45], respectively.

Here, w, h are the width and the height of the output, Y is the output of the last layer, and G is the ground truth phase value, limited in the range $[0, -\pi]$. Additional details about the training of the DNN are provided in Supplement 1, Section 2.

We collected data from six separate experiment runs using training inputs from Faces-LFW or ImageNet and object-to-sensor distances of z_1, z_2, or z_3. These data were used to train six separate DNNs for evaluation.

Figure 2(iv) shows a sample DNN's output at the beginning of its training phase (i.e., with randomly initialized weights), and Fig. 2(v) shows the network output after training for the same example object-raw image pairs. Training each network took ≈20 h using Tensorflow on our GPU. We provide analysis of the trained DNN in Section 3.

Our testing phase consisted of: (1) sampling disjoint examples from the same database (either Faces-LFW or ImageNet) and other databases such as the Modified National Institute of Standards and Technology (MNIST) database, CIFAR [45], Faces-ATT, etc.; (2) using these test examples to modulate the SLM and produce raw intensity images on the camera; (3) passing these intensity images as inputs to our trained DNN; and (4) comparing the output to ground truth.

3. RESULTS AND NETWORK ANALYSIS

The standard method of characterizing neural network training is by plotting the progression of training and test error across training epochs (iterations in the backpropagation algorithm over all examples). These curves are shown in Fig. 4 for our network trained using the ImageNet database and tested using images from: (a) Faces-LFW, (b) a disjoint ImageNet set, (c) images from an English/Chinese/Arabic characters database, (d) the MNIST handwritten digit database, (e) Faces-ATT [46], (f) CIFAR [45], (g) a constant-value "Null" image. Our ImageNet learning curves in Fig. 4(d) show convergence to low value after ∼10 epochs, indicating that our network has not overfit to our training dataset. We plot bar graphs for the mean absolute error (MAE) over test examples in the seven different datasets for each of the three object-to-sensor distances in Fig. 4. Lower MAE was reported for test images with large patches of constant value (characters, digits, Null) as their sparse diffraction patterns were easier for

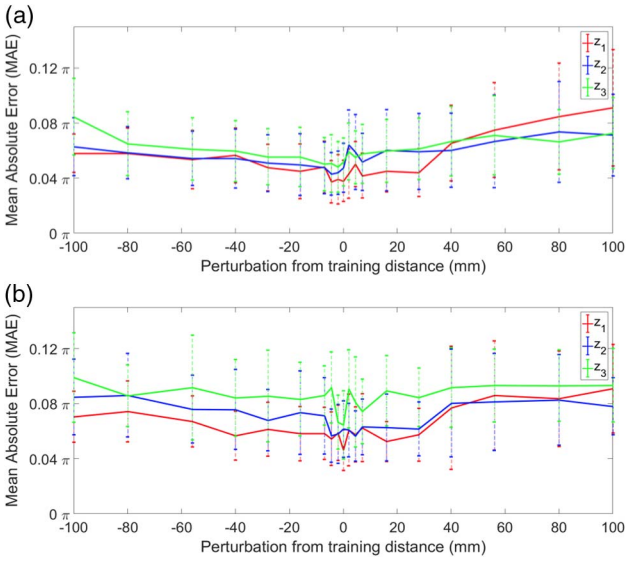

Fig. 6. Quantitative analysis of the sensitivity of the trained deep convolutional neural network to the object-to-sensor distance. The network was trained on (a) Faces-LFW database and (b) ImageNet, and tested on disjoint Faces-LFW and ImageNet sets, respectively. The nominal depths of field for the three corresponding training distances z_1, z_2, z_3 are: $(\mathrm{DOF})_1 = 1.18 \pm 0.1$ mm, $(\mathrm{DOF})_2 = 3.82 \pm 0.2$ mm, and $(\mathrm{DOF})_3 = 7.97 \pm 0.3$ mm, respectively.

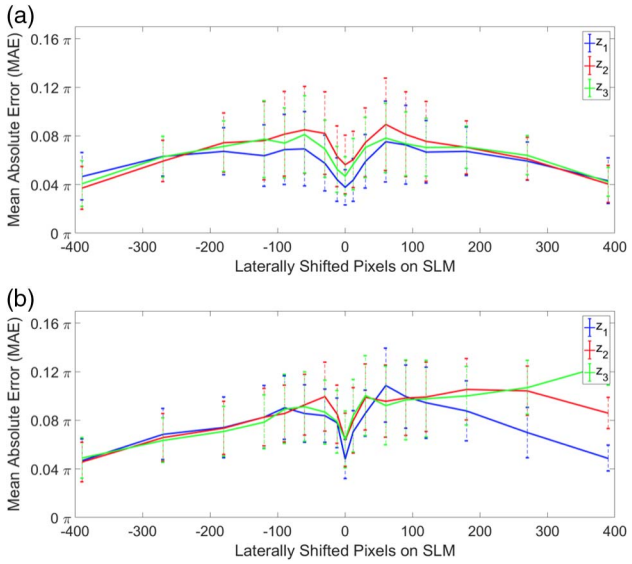

Fig. 7. Quantitative analysis of the sensitivity of the trained deep convolutional neural network to laterally shifted images on the SLM. The network was trained on (a) Faces-LFW database, (b) ImageNet, and tested on disjoint Faces-LFW and ImageNet sets, respectively.

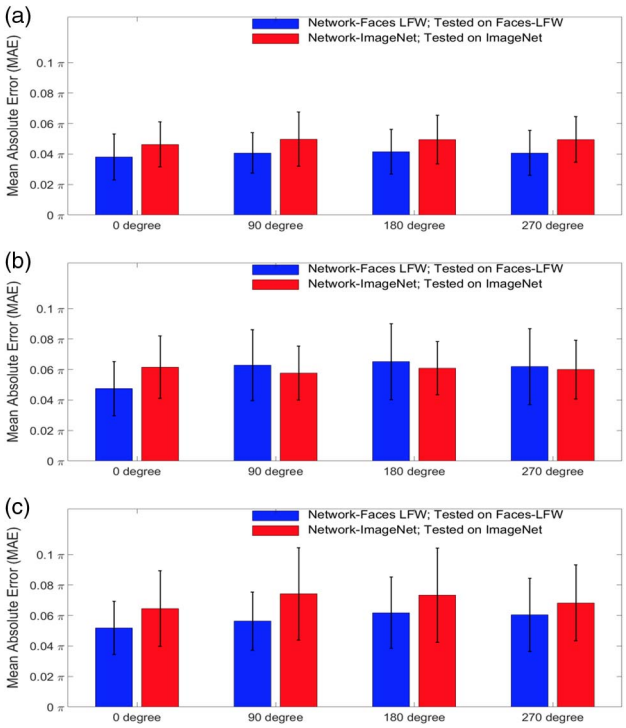

Fig. 8. Quantitative analysis of the sensitivity of the trained deep convolutional neural network to rotation of images on the SLM. The baseline distance on which the network was trained is (a) z_1, (b) z_2, and (c) z_3, respectively.

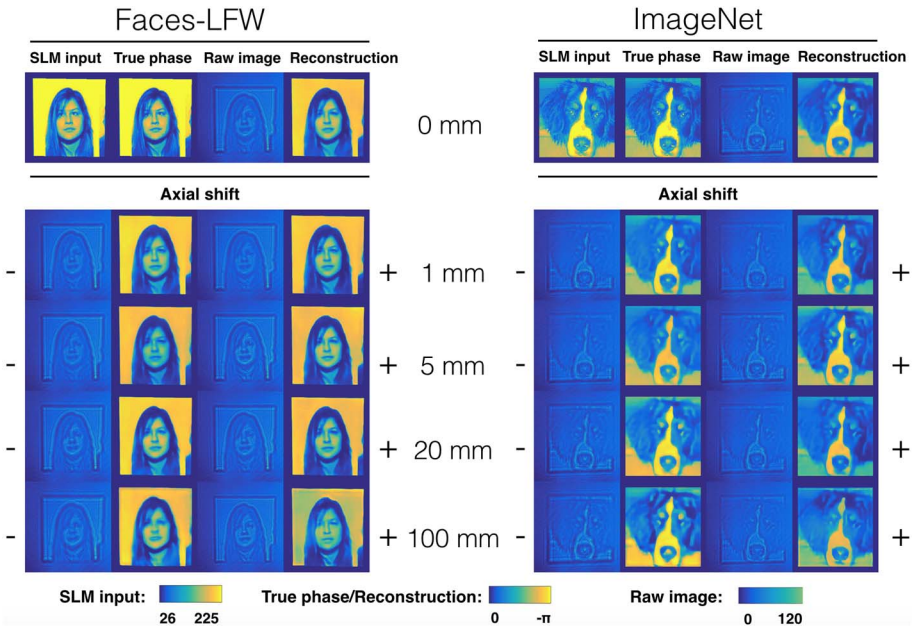

Fig. 9. Qualitative analysis of the sensitivity of the trained deep convolutional neural network to the object-to-sensor distance. The baseline distance on which the network was trained is z_1.

our DNN to invert. Notably, both our bar graphs and learning curves show low test error for the non-trained images, suggesting that our network generalizes well across different domains.

This is an important point and worth emphasizing: despite the fact that our network was trained exclusively on images from the ImageNet database—i.e., images of planes, trains, cars, frogs, artichokes, etc., it is still able to accurately reconstruct images of completely different classes (e.g., faces, handwritten digits, and characters from different languages). This strongly suggests that our network has learned a model of the underlying physics of the imaging system, or at the very least a generalizable mapping of low-level textures between our output diffraction patterns and input images.

A more pronounced qualitative example demonstrating this is shown in the columns (iv) (vii) and (x) of Fig. 5. Here, we trained our network using images exclusively from the Faces-LFW database. Despite this limited training set, the learned network was able to accurately reconstruct images from the ImageNet, handwritten digits, and characters datasets. This is in contrast to results shown in Ref. [33], where an SVM trained on images of faces was able to accurately reconstruct images of faces but not other classes of objects.

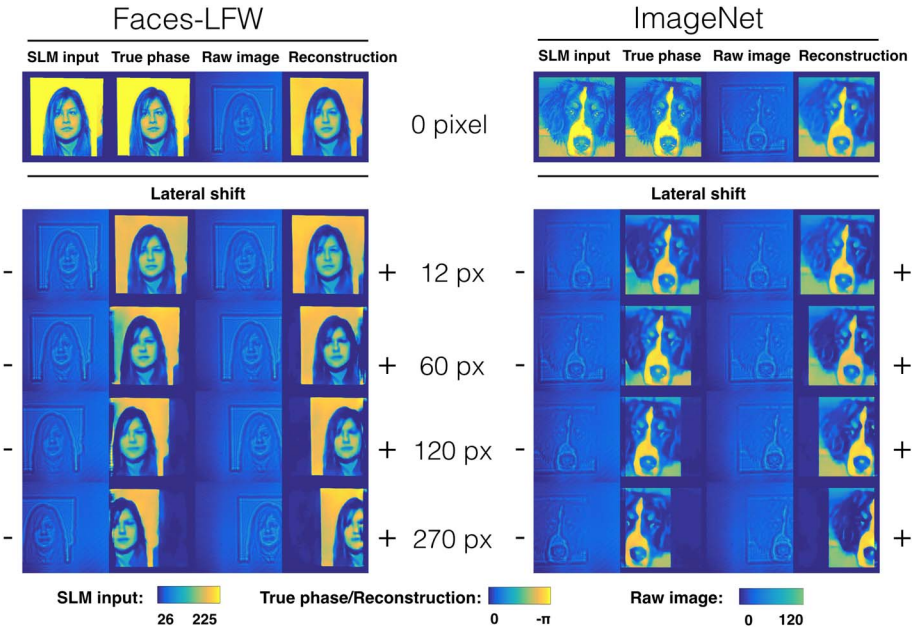

Fig. 10. Qualitative analysis of the sensitivity of the trained deep convolutional neural network to lateral shifts of images on the SLM. The baseline distance on which the network was trained is z_1.

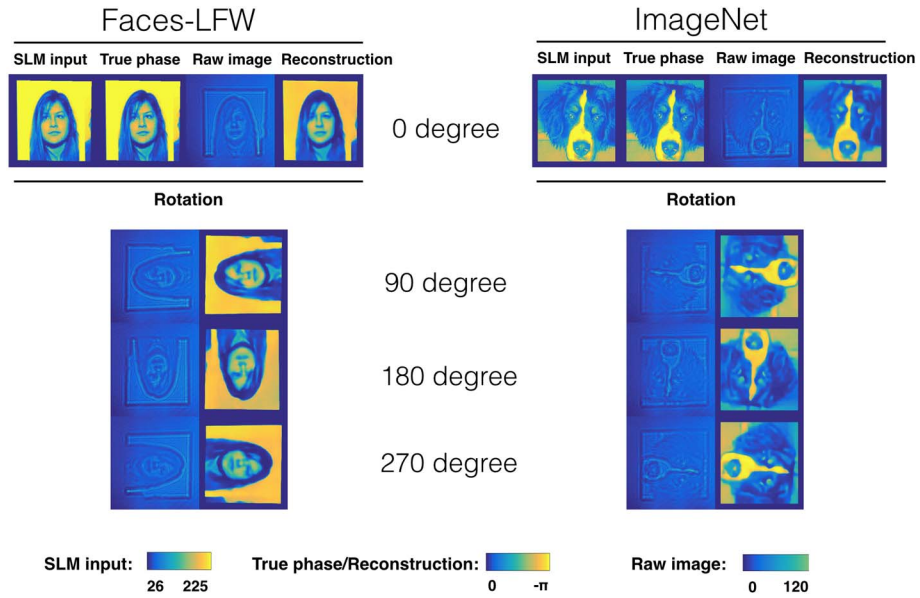

Fig. 11. Qualitative analysis of the sensitivity of the trained deep convolutional neural network to rotation of images in steps of 90. The baseline distance on which the network was trained is z_1.

We tested the robustness of our network to rotation and lateral displacement in the presented test phase objects, as well as axial displacement of the CMOS camera for each DNN relative to the DNN's trained axial locations z_1, z_2, z_3, respectively. Quantitative results of these perturbations are shown in Figs. 6–8, and qualitative results for the networks trained at distance z_1 are shown in Figs. 9–11. Qualitative results for the other two distances are in the supplement. The results show that our trained network has robust to moderate perturbations in sensor displacement and is somewhat shift and rotation invariant. As expected, the reconstruction fails when the displacement is significantly greater (Fig. 12).

To get a sense of what the network has learned, we examined its maximally activated patterns (MAPs), i.e., what types of inputs would maximize network filter response (gradient descent on the input with average filter response as loss function [47]). Our results are shown in Fig. 13, together with the results of analogous analysis of a deblurring network of similar architecture as well as an ImageNet classification DNN. Compared with MAPs of ImageNet and a deblurring network, the MAPs of our phase-retrieval network show two primary differences. First, compared to ImageNet, we observe much finer textures; this is because ImageNet is meant to do classification, a task that requires high-level features to emerge out of learning, whereas our phase network is performing a form of regression. Second, compared with the deblurring network, we observe somewhat finer textures, especially at the shallower layers (although in both cases low-level textures are present). Our interpretation is that both phase retrieval and deblurring require local operations, but of different nature: deblurring converts intensity gradients into sharp edges,

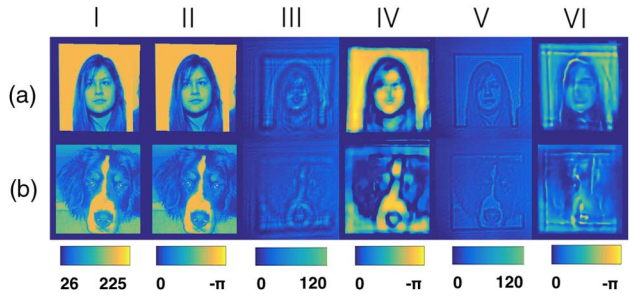

Fig. 12. Failure cases on networks trained on Faces-LFW (row *a*) and ImageNet (row *b*) datasets. (i) Ground truth input; (ii) calibrated phase input to SLM; (iii) raw image on camera; (iv) reconstruction by DNN trained on images at distance z_1 between SLM and camera and tested on images at distance 107.5 cm; (v) raw image on camera; and (vi) reconstruction by network trained on images at distance z_3 between SLM and camera and tested on images at distance 27.5 cm.

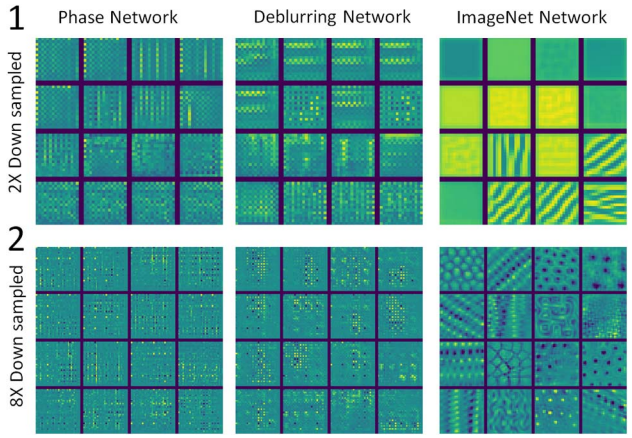

Fig. 13. (1) 16 × 16 inputs that maximally activate the last set of 16 convolutional filters in layer 1 of our phase-retrieval network trained on ImageNet at distance of z_1, a deblurring network, and an ImageNet classification network. The deblurring network was trained on images undergoing motion blur in a random angle within the range [0, 180] degrees and a random blur length in the range [10, 100] pixels. The image is downsampled by a factor of 2 in this layer. (2) 32 × 32 inputs that maximally activate the last set of 16 randomly chosen convolutional filters in layer 3 of our network, the same deblurring network, and the ImageNet classification network. The raw image is downsampled by a factor of 8 in this layer.

whereas phase retrieval converts diffraction rings into phase gradients. The difference between the two cases is not easily discernible by simple visual inspection of the MAPs. The point is that phase retrieval and deblurring share this common locality feature while acting differently for the sake of their respective functions, and both are radically different than classification networks, such as ImageNet.

4. CONCLUSIONS AND DISCUSSION

The architecture presented here was deliberately well controlled, with an SLM creating the phase object inputs to the neural network for both training and testing. This allowed us to quantitatively and precisely analyze the behavior of the learning process. Application-specific training, e.g., replacing the SLM with physical phase objects for more practical applications, we judged beyond the scope of the present work. Other obvious and useful extensions would be to include optics, e.g., a microscope objective for microscopic imaging in the same mode; and to attempt to reconstruct complex objects, i.e., imparting both attenuation and phase delay to the incident light. The significant anticipated benefit in the latter case is that it would be unnecessary to characterize the optics for the formulation of the forward operator—the neural network should "learn" this automatically as well. We intend to undertake such studies in future work.

Funding. Singapore-MIT Alliance for Research and Technology Centre (SMART); Intelligence Advanced Research Projects Activity (IARPA); U.S. Department of Energy (DOE) (DE-FG02-97ER25308).

Acknowledgment. We gratefully acknowledge Ons M'Saad for help with the experiments, Petros Koumoutsakos and Zhengyun Zhang for useful discussions and suggestions, and the anonymous reviewers for careful and constructive criticism.

See Supplement 1 for supporting content.

REFERENCES

1. M. L. Minsky and S. Papert, *Perceptions: An Introduction to Computational Geomry* (MIT, 1969).
2. Y. LeCun, "Generalization and network design strategies," in *Connectionism in Perspective* (Elsevier, 1989), pp. 143–155.
3. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (2012), pp. 1097–1105.
4. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 1–9.
5. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.
6. K. Fukushima, "Cognitron: a self-organizing multilayered neural network," Biol. Cybernet. **20**, 121–136 (1975).
7. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT, 2016).
8. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature **5323**, 533–536 (1986).
9. Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade* (Springer, 2012), pp. 9–48.
10. V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," Nature **518**, 529–533 (2015).
11. D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," Nature **529**, 484–489 (2016).
12. A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox, "Learning to generate chairs, tables and cars with convolutional networks," IEEE Trans. Pattern Anal. Mach. Intell. **39**, 692–705 (2017).
13. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**, 436–444 (2015).
14. Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 415–423.
15. C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European Conference on Computer Vision* (Springer, 2014), pp. 184–199.
16. L. Xu, J. S. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," in *Advances in Neural Information Processing Systems* (2014), pp. 1790–1798.
17. J. Sun, W. Cao, Z. Xu, and J. Ponce, "Learning a convolutional neural network for non-uniform motion blur removal," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 769–777.
18. J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in Neural Information Processing Systems* (2012), pp. 341–349.
19. R. Gerchberg and W. Saxton, "A practical algorithm for the determination of the phase from image and diffraction plane pictures," Optik **35**, 237–246 (1972).
20. J. R. Fienup, "Reconstruction of an object from the modulus of its Fourier transform," Opt. Lett. **3**, 27–29 (1978).
21. J. W. Goodman and R. Lawrence, "Digital image formation from electronically detected holograms," Appl. Phys. Lett. **11**, 77–79 (1967).
22. M. R. Teague, "Deterministic phase retrieval: a Green's function solution," J. Opt. Soc. Am. **73**, 1434–1441 (1983).
23. N. Streibl, "Phase imaging by the transport equation of intensity," Opt. Commun. **49**, 6–10 (1984).
24. M. Bertero and P. Boccacci, *Introduction to Inverse Problems in Imaging* (CRC Press, 1998).
25. A. N. Tikhonov, "On the stability of inverse problems," Dokl. Akad. Nauk SSSR **39**, 195–198 (1943).
26. N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series* (MIT, 1949), Vol. **7**.
27. E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," IEEE Signal Process. Mag. **25**, 21–30 (2008).
28. D. J. Brady, *Optical Imaging and Spectroscopy* (Wiley, 2009).
29. D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via l1 minimization," Proc. Natl. Acad. Sci. USA **100**, 2197–2202 (2003).
30. H. Ji and K. Wang, "Robust image deblurring with an inaccurate blur kernel," IEEE Trans. Image Process. **21**, 1624–1634 (2012).
31. J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," Proc. Natl. Acad. Sci. USA **79**, 2554–2558 (1982).
32. J.-S. Jang, S.-W. Jung, S.-Y. Lee, and S.-Y. Shin, "Optical implementation of the Hopfield model for two-dimensional associative memory," Opt. Lett. **13**, 248–250 (1988).
33. R. Horisaki, R. Takagi, and J. Tanida, "Learning-based imaging through scattering media," Opt. Express **24**, 13738–13743 (2016).
34. S. Popoff, G. Lerosey, R. Carminati, M. Fink, A. Boccara, and S. Gigan, "Measuring the transmission matrix in optics: an approach to the study and control of light propagation in disordered media," Phys. Rev. Lett. **104**, 100601 (2010).
35. S. Popoff, G. Lerosey, M. Fink, A. C. Boccara, and S. Gigan, "Image transmission through an opaque material," arXiv:1005.0532 (2010).
36. U. S. Kamilov, I. N. Papadopoulos, M. H. Shoreh, A. Goy, C. Vonesch, M. Unser, and D. Psaltis, "Learning approach to optical tomography," Optica **2**, 517–522 (2015).

37. Y. Jo, S. Park, J. Jung, J. Yoon, H. Joo, M.-H. Kim, S.-J. Jo, M. C. Choi, S. Y. Lee, and Y. Park, "Holographic deep learning for rapid optical screening of anthrax spores," BioRxiv (2017), 109108.
38. K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," IEEE Trans. Image Process. **26**, 4509–4522 (2017).
39. Y. Rivenson, Y. Zhang, H. Gunaydin, D. Teng, and A. Ozcan, "Phase recovery and holographic image reconstruction using deep learning in neural networks," arXiv:1705.04286 (2017).
40. Y. Rivenson, Z. Gorocs, H. Gunaydin, Y. Zhang, H. Wang, and A. Ozcan, "Deep learning microscopy," arXiv:1705.04709 (2017).
41. A. Sinha, J. Lee, S. Li, and G. Barbastathis, "Lensless computational imaging through deep learning," arXiv:1702.08516 (2017).
42. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein, "Imagenet large scale visual recognition challenge," Int. J. Comput. Vis. **115**, 211–252 (2015).
43. G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: a database for studying face recognition in unconstrained environments," Technical Report (University of Massachusetts, 2007).
44. F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations* (2016).
45. A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Technical Report (University of Toronto, 2009).
46. "AT&T database of faces," Technical Report (AT&T Laboratories Cambridge).
47. M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision* (Springer, 2014), pp. 818–833.