


Approximating the uncertainty of deep learning reconstruction predictions in single-pixel imaging

Ruibo Shang^{1,2}, Mikaela A. O'Brien¹, Fei Wang^{3,4}, Guohai Situ^{3,4,5} & Geoffrey P. Luke¹  [✉]

Single-pixel imaging (SPI) has the advantages of high-speed acquisition over a broad wavelength range and system compactness. Deep learning (DL) is a powerful tool that can achieve higher image quality than conventional reconstruction approaches. Here, we propose a Bayesian convolutional neural network (BCNN) to approximate the uncertainty of the DL predictions in SPI. Each pixel in the predicted image represents a probability distribution rather than an image intensity value, indicating the uncertainty of the prediction. We show that the BCNN uncertainty predictions are correlated to the reconstruction errors. When the BCNN is trained and used in practical applications where the ground truths are unknown, the level of the predicted uncertainty can help to determine whether system, data, or network adjustments are needed. Overall, the proposed BCNN can provide a reliable tool to indicate the confidence levels of DL predictions as well as the quality of the model and dataset for many applications of SPI.

¹Thayer School of Engineering, Dartmouth College, Hanover, NH 03755, USA. ²Department of Bioengineering, University of Washington, Seattle, WA 98195, USA. ³Shanghai Institute of Optics and Fine Mechanics, Chinese Academy of Sciences, Shanghai 201800, China. ⁴Center of Materials Science and Optoelectronics Engineering, University of Chinese Academy of Sciences, Beijing 100049, China. ⁵Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou 310024, China. ✉email: geoffrey.p.luke@dartmouth.edu

Single-pixel imaging (SPI)^{1–3} is a novel imaging technique which uses a single-element photodetector to record the image information instead of using pixel array image sensors. The object is sequentially illuminated by a set of specially designed patterns and the total intensity light for each pattern illumination is collected as a single-pixel value by the photodetector¹. Finally, computational algorithms are applied to reconstruct the object with the sequential intensity collections and the illumination patterns. SPI has many advantages including high speed, broad bandwidth and compact imaging⁴. It also has many applications including remote sensing⁵, holography^{6,7}, optical encryption^{8,9} and tomography¹⁰.

One of the most common reconstruction methods in SPI is sparsity-based optimization which seeks to reconstruct images from incomplete measurements^{11,12} by incorporating the knowledge that most natural images are sparse when the image is transformed into a specific domain. However, the primary drawback is that it is time consuming because of its iterative nature. An image reconstruction task can take up to hours to compute if the scale of the model or scope of the problem is large. Therefore, real-time imaging is infeasible for applications that require pipelined data acquisition and image reconstruction¹³. Besides, the optimal algorithm-specific parameters (i.e., the regularization parameter) generally need to be heuristically determined¹³.

Deep learning (DL)^{14,15} is an emerging and powerful computational imaging tool dramatically improving the state-of-the-art in image reconstruction compared with conventional reconstruction algorithms^{16–22}. It relies on large amounts of training data to automatically learn tasks by finding the optimal weights in each layer of a neural network¹⁵. This is in contrast to iteratively optimizing the image with a specific model in sparsity-based optimization approaches. Therefore, DL is a promising alternative to augment or replace iterative algorithms in sparsity-based optimization¹³. Researchers have applied DL approaches in SPI to improve the quality of the reconstructed images compared with conventional approaches^{13,23–27}. For instance, a DL approach was proposed to predict the image in SPI with an initial guess of the image from conventional approaches as the input to the DL network to further improve the image quality at high compression ratios and noise levels²⁴. End-to-end DL approaches^{13,26} were proposed to predict the image in SPI directly from the raw measurement data without the knowledge of the imaging model and therefore no pre-processing of the raw measurement data is needed.

Generally, the accuracy of the DL predictions in SPI can be quantified by comparing with the ground-truth images¹³ (e.g., calculating mean absolute error (MAE), root mean squared error and structural similarity index (SSIM)²⁸). However, one outstanding challenge is that the ground truth is usually unknown during the prediction stage in many practical applications. Therefore, the accuracy of the DL prediction of a particular image cannot be estimated.

Bayesian convolutional neural networks (BCNNs) have been shown to be an effective approach to approximate the uncertainty with applications including image segmentation²⁹, phase imaging¹⁹, optical metrology³⁰ and image classification³¹. BCNN works on the principle that each pixel in the output image represents the parameter of a probability distribution (e.g., Laplacian or Gaussian distribution), rather than a single intensity value³². Then, the uncertainty can be quantified by Monte Carlo dropout³³ or Deep Ensembles³⁴, for example. BCNNs have many advantages over conventional convolutional neural networks (CNNs). One outstanding advantage is that when the prediction of the image fails in a practical application (i.e., the ground truth is unknown), the BCNN is able to provide an alert on predicted

images with high uncertainty. With the alert, one could consequently make adjustments for better performance.

In this paper, we propose to use a BCNN in SPI to simultaneously predict the image and the pixel-wise uncertainty to quantify the accuracy of the predicted image. We show the BCNN predictions of the image and uncertainty in both simulated and experimental SPI with analysis in details. Overall, these results show that uncertainty approximation can be used to reliably interpret the result of a compressed computational imaging problem.

Results

The BCNN predictions in the simulated SPI trained with the MNIST database. Figure 1a shows a representative ground-truth image in the testing dataset, input images to the network calculated from the LSQR-approximated³⁵ inverse model matrix and BCNN predictions (with all three likelihood functions) at 8×, 16×, 32× and 64× compression ratios. To quantitatively compare the BCNN predictions with the three likelihood functions, the mean and standard deviation of the MAE and SSIM for all the predicted images, and the correlation coefficient (R) between the true absolute error (difference between the ground-truth image and the predicted image) and the predicted uncertainty in the testing dataset at each compression ratio were calculated following Eq. 1 and shown in Fig. 1b–d.

$$R = \frac{1}{N-1} \sum_{i=1}^N \left(\frac{A_i - \mu_A}{\sigma_A} \right) \left(\frac{B_i - \mu_B}{\sigma_B} \right) \quad (1)$$

where A is the true absolute error, B is the predicted uncertainty, μ_A is the mean of A , σ_A is the standard deviation of A , μ_B is the mean of B , σ_B is the standard deviation of B , i is the pixel number and N is the total number of pixels.

Both the qualitative and quantitative results show that the accuracy of the predicted images from the three likelihood functions decreases as the compression ratio increases. This is verified with an increase of the true absolute error in Fig. 1a, an increase of the MAE in Fig. 1b and a decrease of the SSIM in Fig. 1c. This is reasonable since higher compression ratio means higher model ill-posedness which results in solving a more difficult imaging inverse problem¹³. However, the predicted images in BCNN with the Bernoulli-distributed likelihood function are more accurate than those with the Laplacian-distributed and Gaussian-distributed likelihood functions, especially at higher compression ratios. In terms of the predicted uncertainties, the BCNN with the Bernoulli-distributed likelihood function still performs better than the ones with Laplacian-distributed and Gaussian-distributed likelihood functions. In the BCNN with the Bernoulli-distributed likelihood function, the predicted uncertainties generally match well with the true absolute error. The regions of the predicted image from BCNN with larger errors are generally marked with higher uncertainty values in the predicted uncertainty. It can be observed from the true absolute error and predicted uncertainty that most of the higher inaccuracies come from the edges of the image features. However, the predicted uncertainties in the BCNN with Laplacian-distributed and Gaussian-distributed likelihood functions do not match well with the true absolute error. For instance, as shown in Fig. 1a at the 8× compression ratio, higher true absolute errors in the predicted images of BCNN with the two likelihood functions mostly come from the edges of the image features while the predicted uncertainties indicate higher uncertainties not only on the edges but also within the feature regions. The improved performance of the uncertainty predictions with the Bernoulli-distributed likelihood function can also be quantitatively seen in Fig. 1d with a generally higher

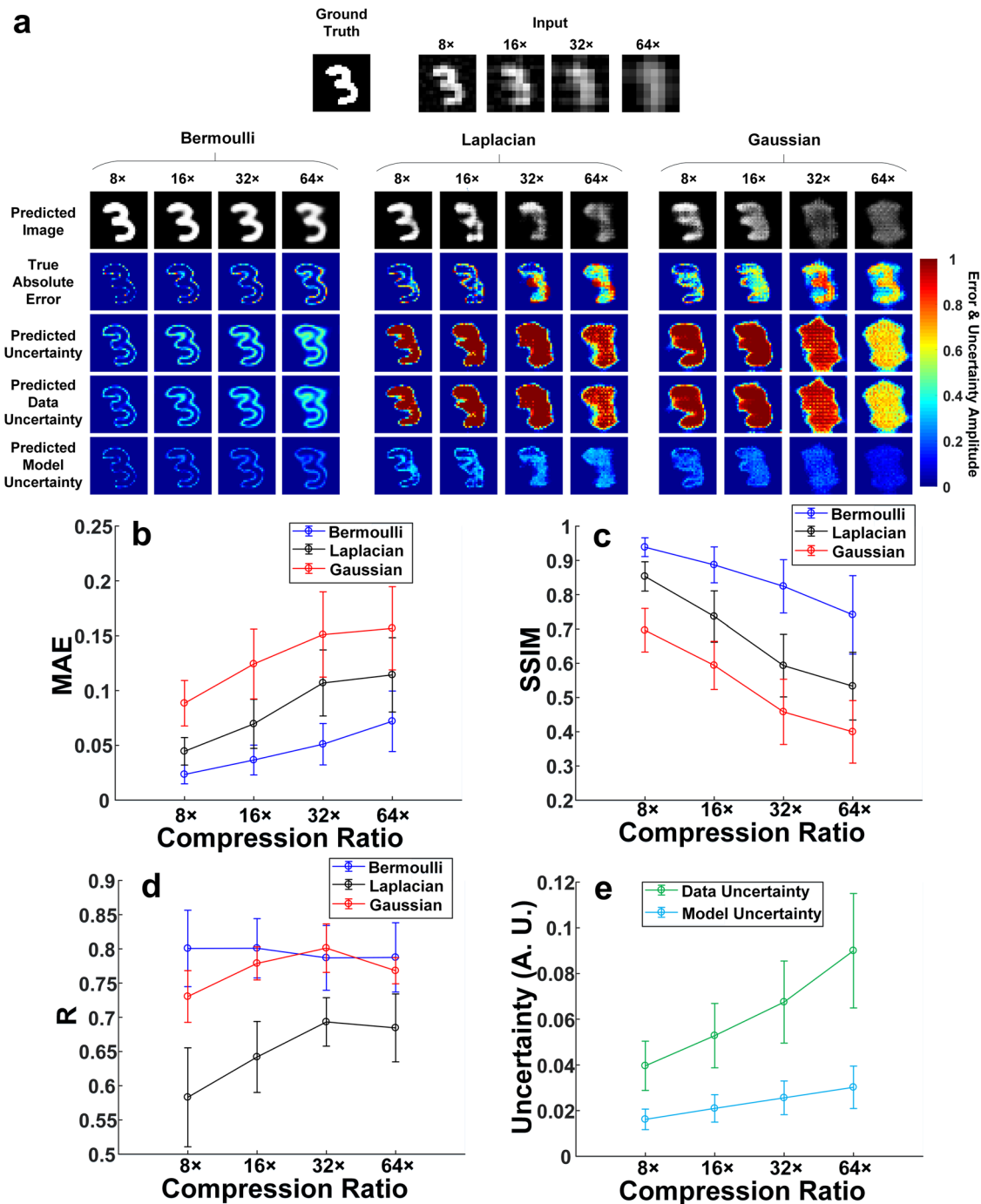


Fig. 1 The BCNN predictions in the simulated SPI trained with the MNIST database. **a** A representative ground-truth image in the testing dataset, input images to the BCNN calculated from the LSQR-approximated inverse model matrix and the BCNN predictions with Bernoulli-distributed, Laplacian-distributed and Gaussian-distributed likelihood functions at the 8x, 16x, 32x and 64x compression ratios. **b** The MAEs of the predicted images in BCNN with the three likelihood functions at the four compression ratios. **c** The SSIMs of the predicted images in BCNN with the three likelihood functions at the four compression ratios. **d** The correlation coefficient, R, between the predicted uncertainty and the absolute error of each pixel the predicted images reconstructed with the three likelihood functions at the four compression ratios. **e** Averaged pixel values of the predicted data and model uncertainties in the testing dataset with the Bernoulli-distributed likelihood function at the four compression ratios. The error bars represent the standard deviation of the corresponding parameters from 100 testing images.

correlation coefficient between the predicted uncertainty and the true absolute error.

We also explored the effect of noise on both the image and uncertainty predictions in BCNN (Supplementary Note 1). The results show that the performance of BCNN decreases as the signal-to-noise ratio (SNR) decreases from 25 dB to 0 dB.

However, the fidelity of the corresponding result is still at a high level even if the data SNR is as low as 0 dB, suggesting good robustness to noise.

In summary, for the MNIST database³⁶, the BCNN with the Bernoulli-distributed likelihood function performs the best among the BCNNs with three distribution likelihood functions.

The BCNNs with the Laplacian-distributed and Gaussian-distributed likelihood functions are not suitable for the MNIST database. This is reasonable since the modified images in the MNIST database are binary, which fits with the Bernoulli distribution. In addition, it is observed that the data uncertainty is dominant over the model uncertainty. This effect becomes more pronounced at higher compression ratios. This can be shown quantitatively by the averaged pixel values in the predicted data and model uncertainties in the testing dataset with the Bernoulli-distributed likelihood function in Fig. 1e. We hypothesize that this comes from the compressed nature of the measurement data in the training set of the MNIST database.

Effect of the physics-prior based preprocessor and uncertainty estimation on network performance. For the BCNN predictions shown in Fig. 1, a pre-processing step was used to convert the inputs of the neural network from measurement domain into image domain. In this section, we sought to explore the effect of the physics-prior based preprocessor to the BCNN performance. We also compared the performance of BCNN with the conventional CNN, which does not have the uncertainty prediction. We denote BCNN as the BCNN with the physics-prior based preprocessor, End-To-End BCNN as the BCNN with the one-dimensional (1D) raw measurement data as the network input, CNN as the CNN with the physics-prior based preprocessor but without the uncertainty prediction function, and End-To-End CNN as the CNN without either the physics-prior based preprocessor or the uncertainty prediction function. To solve the dimension mismatch between the 1D raw measurement data and the two-dimensional image, a fully-connected layer (together with reshape and permute layers) was added in between the input layer and the first convolutional layer of the BCNN to generate End-To-End BCNN. CNN and End-To-End CNN have the same network structures as BCNN and End-To-End BCNN respectively, except that there is no uncertainty prediction incorporated in the loss function. The Bernoulli-distributed likelihood function was used in BCNN and End-To-End BCNN. The training and validation curves are shown in Supplementary Fig. 3.

Figure 2a shows the ground-truth image, the input images for BCNN and CNN, the 1D raw measurement data as the input to End-To-End BCNN and End-To-End CNN, and predictions from BCNN, CNN, End-To-End BCNN and End-To-End CNN at 8 \times , 16 \times , 32 \times and 64 \times compression ratios. To quantitatively compare BCNN, CNN, End-To-End BCNN and End-To-End CNN, the mean and standard deviation of the MAE and SSIM for all the predicted images were calculated and shown in Fig. 2b, c. Figure 2d, e show the averaged pixel values of the predicted model and data uncertainties in BCNN and End-To-End BCNN in the testing dataset at the four compression ratios. The results show that BCNN and CNN have comparable performance on image predictions in terms of MAE and SSIM (Fig. 2b, c), which means that the extra uncertainty predictions in BCNN do not affect its image predictions compared to the conventional CNN. The results also show that BCNN and CNN have better performance in image predictions than End-To-End BCNN and End-To-End CNN in terms of MAE and SSIM. The reason for this outperformance is that BCNN and CNN incorporate physics priors to obtain the initial-guess images as the network inputs to reduce the uncertainty from the data, thus improving the image predictions. This can also be verified in Fig. 2e where BCNN has lower data uncertainties than End-To-End BCNN at all the four compression ratios. Besides, BCNN and End-To-End BCNN have roughly the same model uncertainties at all the four compression ratios since they have similar network structures.

The BCNN predictions in the simulated SPI trained with the STL-10 database. In this section, we explore the BCNN performances with the three likelihood functions in the simulated SPI with a more challenging task where the STL-10 database³⁷ with more complexed image features is used for training and predictions. Figure 3a shows a representative ground-truth image in the testing dataset, input images to the network calculated from the LSQR-approximated inverse model matrix and Fig. 3b shows BCNN predictions (with all three likelihood functions) at 2 \times , 4 \times , 8 \times and 16 \times compression ratios. The mean and standard deviation of the MAE and SSIM for all the predicted images, and the correlation coefficient between the true absolute error and the predicted uncertainty in the testing dataset with the three likelihood functions at each compression ratio were calculated and shown in Supplementary Table 1.

The results in Fig. 3 and Supplementary Table 1 show that the accuracy of the predicted images from the three likelihood functions decreases as the compression ratio increases, which is reasonable since higher compression ratio means higher model ill-posedness which results in solving a more difficult imaging inverse problem¹³. Besides, the prediction of the images in BCNNs with the three likelihood functions performs close to each other as shown qualitatively in the predicted-image rows in Fig. 3b and quantitatively in terms of MAE and SSIM in Supplementary Table 1. The predicted uncertainties in BCNNs with the Laplacian-distributed and Gaussian-distributed likelihood functions match well with the true absolute error since the regions where the predicted image from BCNN has larger errors are generally marked with higher uncertainty values in the predicted uncertainty. However, the predicted uncertainties in BCNN with the Bernoulli-distributed likelihood function are much worse as shown qualitatively in Fig. 3b where the low true-absolute-error pixels are marked with higher uncertainty values in the predicted uncertainty instead, and quantitatively in Supplementary Table 1 where the correlation coefficient R between the true absolute error and the predicted uncertainty from the BCNN with the Bernoulli-distributed likelihood function are much lower than those with the Laplacian-distributed and Gaussian-distributed likelihood function. This is reasonable since the loss function for the Bernoulli distribution in Eq. 16 also minimizes the error between the mean of the pixel distribution and the ground truth. Therefore, the predictions of the images perform close to those using the Laplacian-distributed and Gaussian-distributed likelihood function. The uncertainty prediction, however, is dependent on both the predicted mean and the predicted standard deviation. In the case of the Bernoulli distribution and STL-10 dataset, the predicted standard deviation denotes how far the pixel value is from 1 or 0 since it expects a binary image, while the images in the STL-10 database are in gray scale. In this case, the data uncertainty and the overall uncertainty calculated from Eq. 21 will be wrong. The model uncertainty which is the variance of the predicted mean with Monte Carlo Dropout, however, is reasonable since the predicted mean is correct. Therefore, it indicates that when using BCNN to make predictions in SPI with the STL-10 database, the Laplacian-distributed and Gaussian-distributed likelihood functions can be used while the Bernoulli-distributed likelihood function is not suitable. It is also observed that the predicted uncertainty and the true absolute error from the Laplacian-distributed and Gaussian-distributed likelihood functions are only modestly correlated. The modest correlation comes from the fact that not all areas of higher uncertainty necessarily have high error. They merely point out pixels where high errors are likely to occur. A perfect correlation would indicate that perfect reconstruction is possible. Besides, similar to the observations in the other simulations, it can still be observed from the true absolute error and predicted

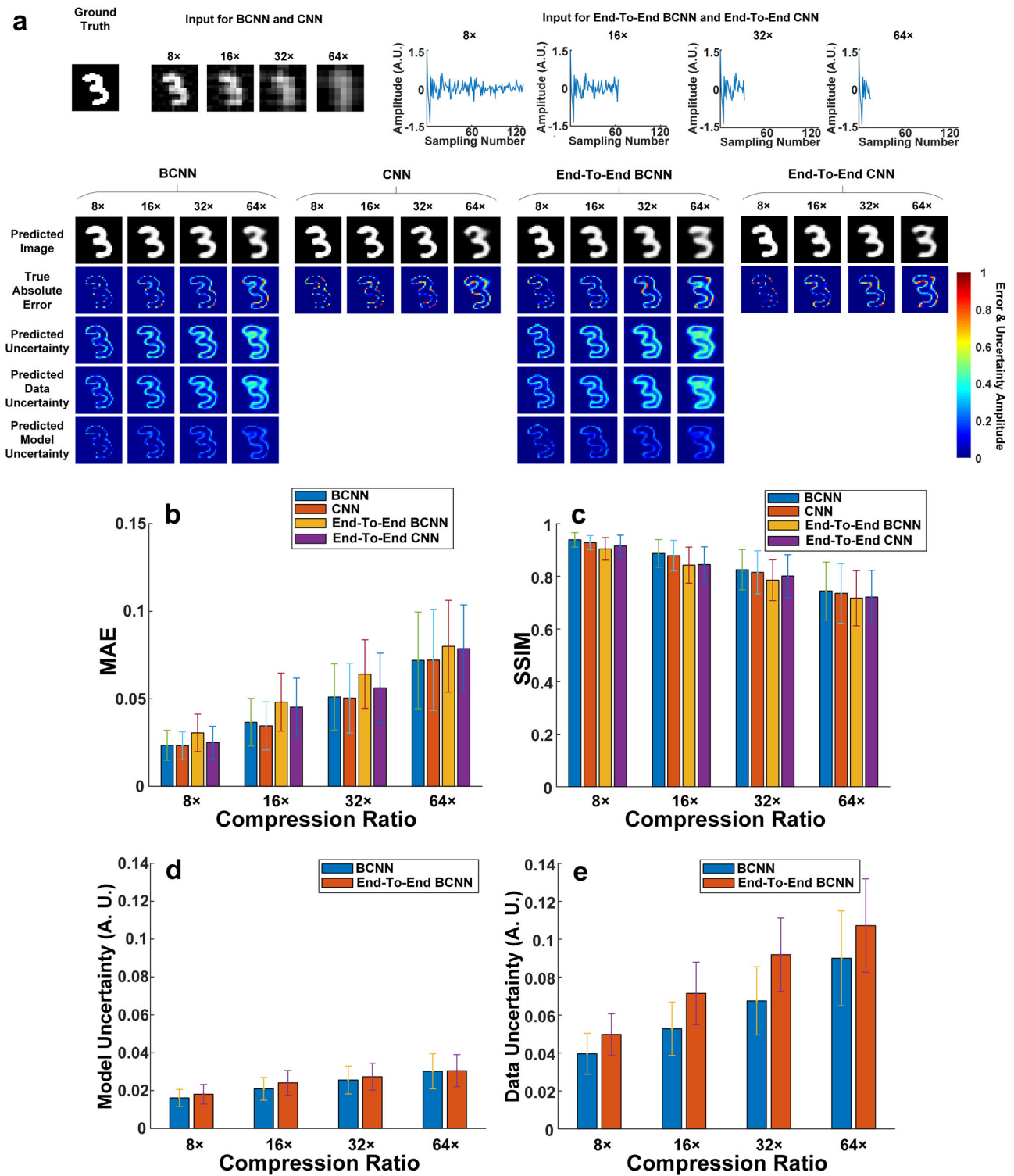


Fig. 2 Comparisons among BCNN, CNN, End-To-End BCNN and CNN in the simulated SPI trained with the MNIST database. **a** A representative ground-truth image in the testing dataset, input images to the BCNN and CNN calculated from the LSQR-approximated inverse model matrix, 1D raw measurement data as the input to End-To-End BCNN and End-To-End CNN, and the predictions from BCNN, CNN, End-To-End BCNN and End-To-End CNN at the 8x, 16x, 32x and 64x compression ratios. **b** The MAEs of the predicted images in BCNN, CNN, End-To-End BCNN and End-To-End CNN at the four compression ratios. **c** The SSIMs of the predicted images in BCNN, CNN, End-To-End BCNN and End-To-End CNN at the four compression ratios. **d** Averaged pixel values of the predicted model uncertainties in BCNN and End-To-End BCNN in the testing dataset at the four compression ratios. **e** Averaged pixel values of the predicted data uncertainties in BCNN and End-To-End BCNN in the testing dataset at the four compression ratios. The error bars represent the standard deviation of the corresponding parameters from 100 testing images.

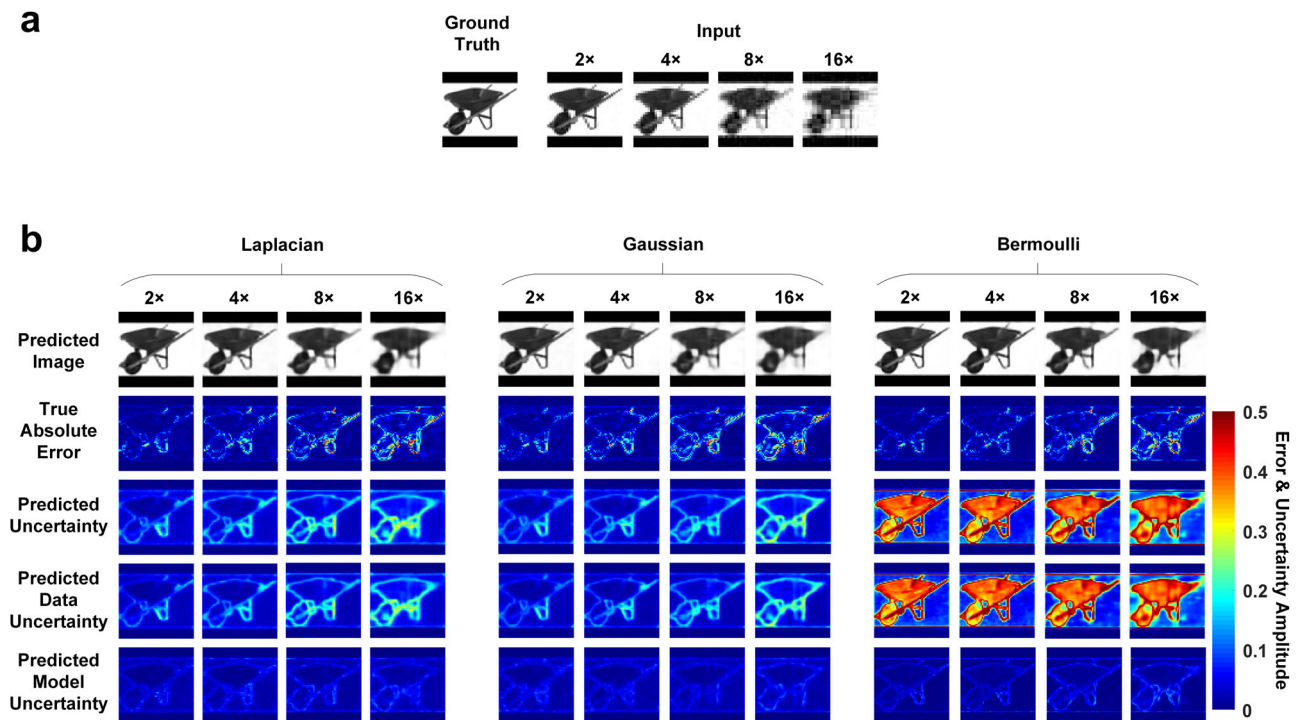


Fig. 3 The results of BCNN with Laplacian-distributed, Gaussian-distributed and Bernoulli-distributed likelihood functions in simulated SPI with STL-10 dataset at 2 \times , 4 \times , 8 \times and 16 \times compression ratios. **a** A representative ground-truth image in the testing dataset, input images to the BCNN calculated from the LSQR-approximated inverse model matrix at the 2 \times , 4 \times , 8 \times and 16 \times compression ratios. **b** BCNN predictions with Laplacian-distributed, Gaussian-distributed and Bernoulli-distributed likelihood functions at the 2 \times , 4 \times , 8 \times and 16 \times compression ratios.

uncertainty that most of the higher inaccuracies come from the edges of the image features, and that the data uncertainty is dominant over the model uncertainty.

We also quantitatively compared BCNN, CNN, End-To-End BCNN and End-To-End CNN with the STL-10 database in the simulated SPI at the 4 \times compression ratio. The Laplacian-distributed likelihood function was used in BCNN and End-To-End BCNN. The results are shown in Supplementary Fig. 4. Similar to the comparisons of the four neural networks with the MNIST database, BCNN and CNN have similar performance on image predictions in terms of MAE and SSIM, which means that the extra uncertainty predictions in BCNN do not affect its image predictions compared to the conventional CNN. The results also show that BCNN and CNN have better overall performance in image predictions than End-To-End BCNN and End-To-End CNN in terms of MAE and SSIM.

Thus far, the BCNN was applied separately to either the MNIST and STL-10 database. We also explored a different training strategy where the BCNN was trained with a mixture of the MNIST and STL-10 databases (Hybrid Training) with either Laplacian-distributed or Bernoulli-distributed likelihood functions. We quantitatively compared its performance with the one trained on the two databases separately (Separate Training) in Supplementary Note 2. The results show that Separate Training has better overall performance in image and uncertainty predictions than Hybrid Training.

Experimental results. Figure 4a shows representative ground-truth images from the testing dataset. Input images and the predictions of BCNN at 16 \times and 64 \times compression ratios are shown in Fig. 4b, c. The BCNN provides reasonably good predicted images in SPI at both 16 \times and 64 \times compression ratios. However, the images at 64 \times are in poorer quality than those at 16 \times . This can be visualized from the true absolute errors in

Fig. 4b, c. Quantitative results of BCNN predictions are shown in Fig. 5c–e. In terms of the predicted images, the performance of the BCNN decreases as the compression ratio increases from 16 \times to 64 \times . However, it still remains at a good level with an MAE lower than 0.1 and an SSIM higher than 0.6. The performance of the BCNN in terms of the predicted uncertainty remains at almost the same level at the two compression ratios, showing its great robustness. Visually, the predicted uncertainty generally matches well with the true absolute error in Fig. 4b, c. The regions where the predicted image from BCNN has larger errors are generally marked with higher uncertainty values in the predicted uncertainty. It can still be observed from the true absolute error and predicted uncertainty that most of the higher inaccuracies come from the edges of the image features. Again, the data uncertainty is dominant over the model uncertainty due to the compressed nature and noise in the training dataset.

We also quantitatively compared BCNN, CNN, End-To-End BCNN and End-To-End CNN in this experimental SPI with the MNIST database. Figure 5a shows a representative ground-truth image, the input images for BCNN and CNN, the 1D raw measurement data as the input to End-To-End BCNN and End-To-End CNN, and predictions from BCNN, CNN, End-To-End BCNN and End-To-End CNN at 16 \times and 64 \times compression ratios. Figure 5b shows a representative ground-truth image out of the MNIST database, the input images for BCNN and CNN, the 1D raw measurement data as the input to End-To-End BCNN and End-To-End CNN, and predictions from BCNN, CNN, End-To-End BCNN and End-To-End CNN at 16 \times and 64 \times compression ratios. It shows that BCNN has good generalization performance in the experimental SPI. To quantitatively compare BCNN, CNN, End-To-End BCNN and End-To-End CNN, the mean and standard deviation of the MAE and SSIM for all the predicted images were calculated and are shown in Fig. 5c, d. Figure 5e shows the mean and standard deviation of the

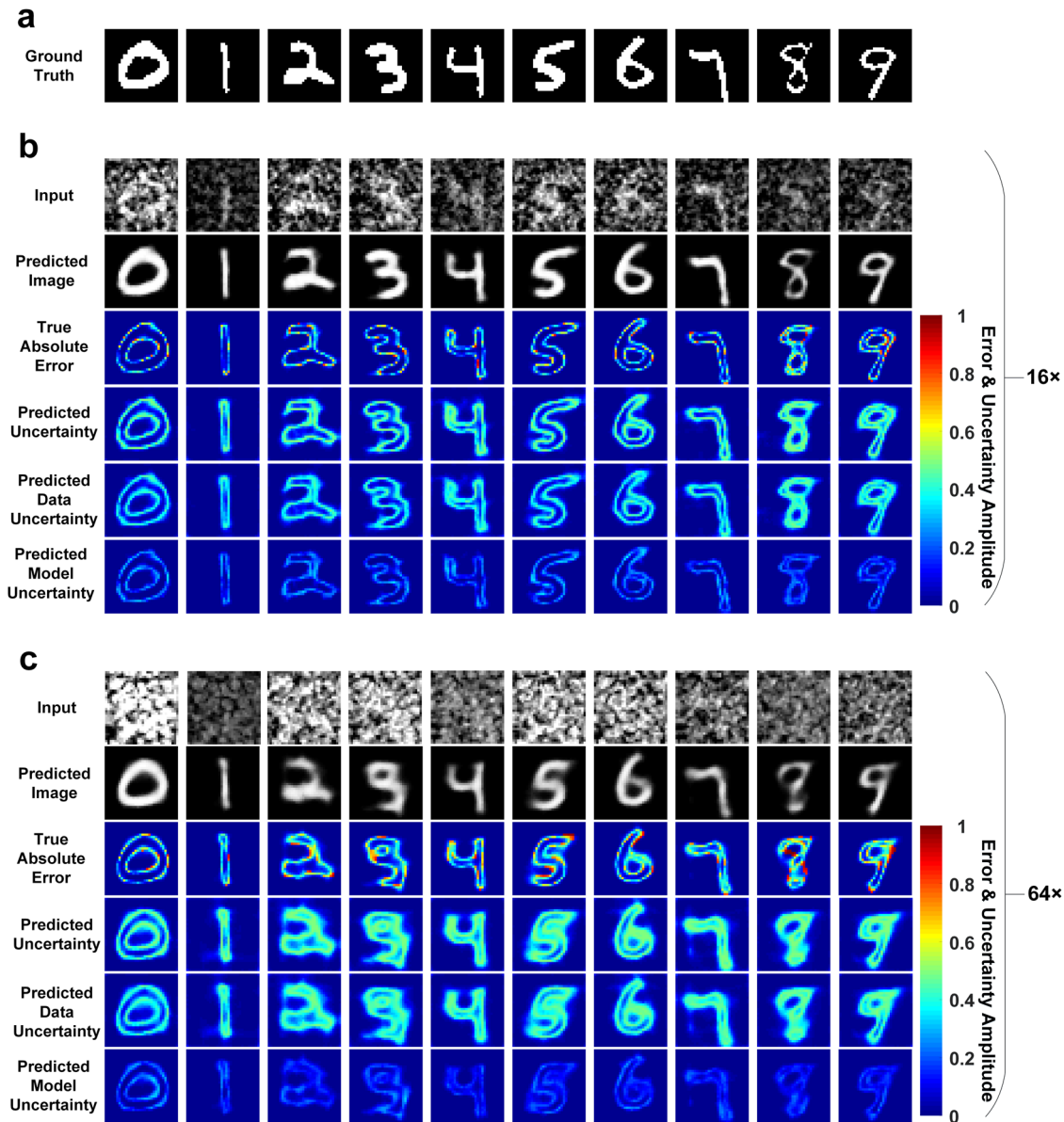


Fig. 4 Experiment results with BCNN in SPI with the MNIST dataset at 16× and 64× compression ratios. **a** Ten representative ground-truth images from the MNIST testing dataset. **b** Input images and predictions of BCNN in SPI at 16× compression ratio. **c** Input images and predictions of BCNN in SPI at 64× compression ratio.

correlation coefficient R in BCNN and End-To-End BCNN for all the predicted images. The training and validation curves are shown in Supplementary Fig. 5. The results show that CNN has only slightly better performance than the BCNN (Fig. 5c, d), which means that the extra uncertainty predictions in BCNN do not appreciably affect their image predictions compared to the conventional CNN. The results also show that BCNN and CNN have similar performance in image predictions compared to End-To-End BCNN and End-To-End CNN in terms of MAE and SSIM, which is slightly different from the corresponding conclusion in the simulated case. The reason for the difference is that random grayscale patterns were used in the experiments instead of Russian-Doll (RD) Hadamard patterns used in the simulation, leading to a more ill-posed inverse problem. Therefore, even though BCNN and CNN incorporate physics priors to obtain the initial-guess images as the network inputs, the data uncertainty is not reduced, thus not improving the image predictions. Besides, BCNN and End-To-End BCNN have

roughly the same correlation coefficient R at both compression ratios as shown in Fig. 5e, indicating that they have similar performance in uncertainty predictions.

Discussion

The BCNN is proposed for uncertainty approximation in SPI with Bernoulli-distributed, Laplacian-distributed or Gaussian-distributed likelihood functions with the MNIST and STL-10 databases. First, the BCNNs with the three distribution likelihood functions were compared in simulated SPI with the MNIST database at varying compression ratios and the Bernoulli-distributed likelihood function was proved to be the most appropriate among the three functions. Second, the robustness of BCNN to noise from the measurement data was studied with the Bernoulli-distributed likelihood function and the MNIST database (Supplementary Note 1). Third, the three likelihood functions were compared in BCNN in simulated SPI with STL-10 dataset and the Laplacian-distributed and Gaussian-distributed

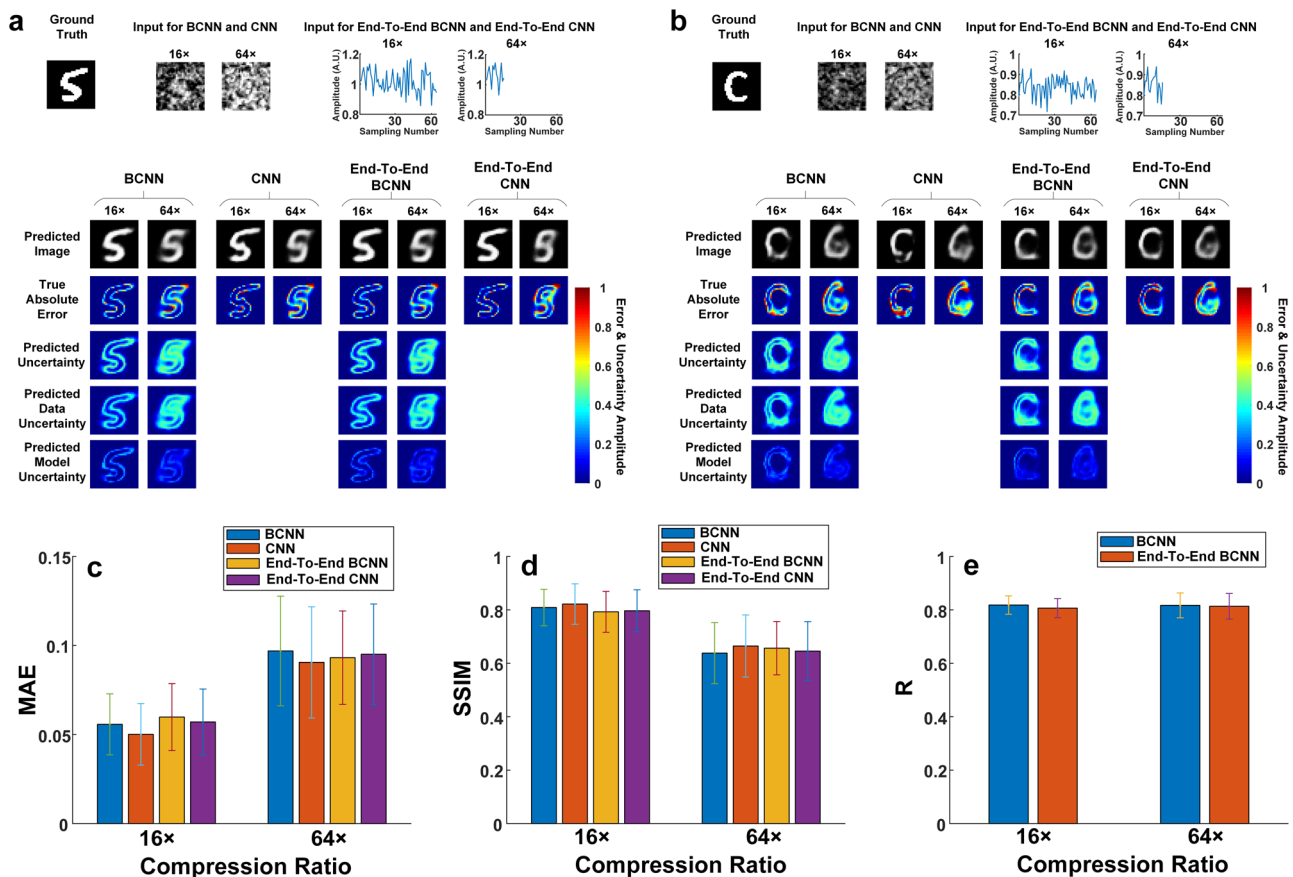


Fig. 5 Comparisons among BCNN, CNN, End-To-End BCNN and End-To-End CNN in the experimental SPI trained with the MNIST database. **a** A representative ground-truth image in the testing dataset, input images to the BCNN and CNN calculated from the LSQR-approximated inverse model matrix, 1D raw measurement data as the input to End-To-End BCNN and End-To-End CNN, and the predictions from BCNN, CNN, End-To-End BCNN and End-To-End CNN at the 16× and 64× compression ratios. **b** A ground-truth image out of the testing dataset in the MNIST database, input images to the BCNN and CNN calculated from the LSQR-approximated inverse model matrix, 1D raw measurement data as the input to End-To-End BCNN and End-To-End CNN, and the predictions from BCNN, CNN, End-To-End BCNN and End-To-End CNN at the 16× and 64× compression ratios. **c** The MAEs of the predicted images in BCNN, CNN, End-To-End BCNN and End-To-End CNN at the two compression ratios. **d** The SSIMs of the predicted images in BCNN, CNN, End-To-End BCNN and End-To-End CNN at the two compression ratios. **e** The correlation coefficient, R , between the predicted uncertainty and the true absolute error of each pixel the predicted images reconstructed in BCNN and End-To-End BCNN at the two compression ratios. The error bars represent the standard deviation of the corresponding parameters from 100 testing images.

likelihood functions were shown to be equivalent and both better than the Bernoulli-distributed likelihood function in this application. Fourth, different training strategies were compared in Supplementary Note 2. Fifth, in experiments, the BCNN with Bernoulli-distributed likelihood functions was used and verified in experimental SPI with the MNIST dataset at 16× and 64× compression ratios. In all the simulations and experiments, the performances of BCNN, CNN, End-To-End BCNN and End-To-End CNN were quantitatively evaluated to study the effect of the physics-prior based preprocessor and the uncertainty estimation on network performance.

BCNNs have advantages over conventional CNNs. As shown in Figs. 1–5, BCNNs not only predict the image as conventional CNNs do but also provide a reliability assessment to indicate the pixel-wise uncertainties of DL predictions as well as the quality of the model and dataset with model uncertainty and data uncertainty respectively. As the quality of the predicted images decreases, the corresponding uncertainty values increase to indicate this change of the quality. For a specific predicted image, the pixel-wise uncertainty prediction tells the error of each pixel in the predicted image and highlights where the large errors occur in the predicted image. This is especially useful to evaluate the prediction of the neural network when the ground truth is

unknown in many practical applications, and the level of the predicted uncertainty can be used to determine whether some adjustments in the imaging system, training data, and/or network architecture are needed.

However, several aspects of this work still need further exploration. First, how to choose the optimal probability-distributed likelihood function in BCNN efficiently is a problem that needs to be explored. In this work, we trained the BCNN with the potential probability-distributed likelihood functions and then compared the results to find the optimal one. However, the drawback is that it is time consuming. Based on our experience, the Bernoulli-distributed likelihood function works well on binary images, and the Laplacian-distributed and Gaussian-distributed likelihood functions work equally well on natural grayscale images. We would like to propose a method to find the optimal distribution before training using the dataset statistics and network architecture. Second, it is observed that the data uncertainty is dominant over the model uncertainty in both simulations and experiments due to the compressed nature and noise in the measurement data in the training dataset. We would like to search for more advanced pre-processing approaches to decrease the uncertainty stemming from the measurement data. Third, we would like to explore ways to use the predicted uncertainty as a

feedback to optimize the BCNN structures to further decrease the uncertainty of the results, or to decrease the network complexity without a loss in performance.

In summary, the proposed BCNN enables uncertainty approximation in SPI. It is a reliable tool in diverse applications in SPI where the confidence on the predicted images needs to be approximated.

Methods

Mathematical basis of SPI. Here, we focus our discussion on two-dimensional imaging in SPI. We denote the object as $O(x, y)$, and the set of patterns used to illuminate the object as $P_m(x, y)$, where $m = 1, 2, \dots, M$ (M is the total number of patterns). The 1D signal acquired in SPI can be written as,

$$I_m = \int P_m(x, y)O(x, y)dx dy \tag{2}$$

When the signal is sampled at discrete pixel locations, Eq. (2) can be written as,

$$I_m = \sum_{a=1}^{N_x} \sum_{b=1}^{N_y} P_m(x_a, y_b)O(x_a, y_b) \tag{3}$$

where x_a and y_b denote discrete pixel locations. N_x and N_y denote the total pixel numbers in x and y dimensions.

Equation (3) represents a linear model, and can be written as matrix multiplication by

$$g = \mathbf{H}f + n \tag{4}$$

where f is the vectorized version of the object O (the dimension of f is $N_x N_y \times 1$), g is the raw measurement (the dimension of g is $M \times 1$), n is the noise and \mathbf{H} is the forward operator, where the m^{th} row of \mathbf{H} contains the vectorized version of the illumination pattern P_m (the dimension of \mathbf{H} is $M \times N_x N_y$).

The inverse problem of Eq. (4) is ill-posed due to the compression property of SPI. Therefore, a regularized optimization approach is usually used in SPI to incorporate additional knowledge about the image by adding a regularization term,

$$\hat{f} = \arg \min_f \{ \|\mathbf{H}f - g\|_2^2 + \lambda \phi(f) \} \tag{5}$$

where ϕ is the regularization operator and λ is the regularization parameter. $\|\mathbf{H}f - g\|_2^2$ is the fidelity term and $\phi(f)$ is the regularization term. Common regularization domains include spatial, edge, and wavelet domains. Equation (5) can be solved by iterative optimization approaches or deep learning approaches.

Bayesian networks for uncertainty approximation. As opposed to conventional convolutional neural networks where the weights are deterministic after training, BCNNs use distributions over the network parameters to replace the deterministic weights in the network³². This probabilistic property of BCNN come from the stochastic (random) processes in the network such as dropout³⁸, weight initialization³⁹ etc. Suppose the training dataset is denoted as $(X, Y) = \{x_n, y_n\}_{n=1}^N$ with X and Y representing the network inputs and ground-truth images, respectively. N is the total number of images in the training dataset. To approximate the variability of the prediction y given a specific input $x_{test,t}$ in the testing dataset $(X_{test}, Y_{test}) = \{x_{test,t}, y_{test,t}\}_{t=1}^T$ (T is the total number of images in the testing dataset), we use the predictive distribution $p(y|x_{test,t}, X, Y)$ over all possible learned weights (with marginalization)³³:

$$p(y|x_{test,t}, X, Y) = \int p(y|x_{test,t}, W)p(W|X, Y)dW \tag{6}$$

where $p(y|x_{test,t}, W)$ denotes the predictive distribution that

includes all possible output predictions given the learned weights W and the input $x_{test,t}$ from the testing dataset. It can be understood as data uncertainty¹⁹. $p(W|X, Y)$ denotes all possible learned weights given the training dataset, which can be understood as model uncertainty¹⁹.

To model the data uncertainty, we need to define the probability distribution of the BCNN outputs with a specific likelihood function. In this paper, we choose the multivariate Laplacian-distributed, Gaussian-distributed and Bernoulli-distributed likelihood functions to model the data uncertainty.

- (a) We define the multivariate Laplacian-distributed likelihood function as:

$$P_{Laplacian}(y|x, W) = \prod_{m=1}^M P_{Laplacian}(y^m|x, W) \tag{7}$$

$$P_{Laplacian}(y^m|x, W) = \frac{1}{2\sigma^m} \exp\left(-\frac{|y^m - \mu^m|}{\sigma^m}\right) \tag{8}$$

where m denotes the m^{th} pixel in the BCNN output image, M denotes the total number of pixels in the BCNN output image, and μ^m and σ^m denote the mean and standard deviation of the m^{th} pixel in the BCNN output image, respectively.

By taking logarithm and negative operations on Eq. (7), the loss function $L_{Laplacian}(W|x_n, y_n)$ for the Laplacian-distributed likelihood function given the training data pair (x_n, y_n) is:

$$L_{Laplacian}(W|x_n, y_n) = \frac{1}{M} \sum_{m=1}^M \left[\frac{|y_n^m - \mu_n^m|}{\sigma_n^m} + \log(2\sigma_n^m) \right] \tag{9}$$

- (b) For multivariate Gaussian-distributed likelihood function, we define:

$$P_{Gaussian}(y|x, W) = \prod_{m=1}^M P_{Gaussian}(y^m|x, W) \tag{10}$$

$$P_{Gaussian}(y^m|x, W) = \frac{1}{\sqrt{2\pi}\sigma^m} \exp\left[-\frac{(y^m - \mu^m)^2}{2(\sigma^m)^2}\right] \tag{11}$$

where the denotations are the same as those in Eqs. (7) and (8).

By taking logarithm and negative operations on Eq. (10), the loss function $L_{Gaussian}(W|x_n, y_n)$ for the Gaussian-distributed likelihood function given the training data pair (x_n, y_n) is:

$$L_{Gaussian}(W|x_n, y_n) = \frac{1}{M} \sum_{m=1}^M \left[\frac{(y_n^m - \mu_n^m)^2}{2(\sigma_n^m)^2} + \log(\sqrt{2\pi}\sigma_n^m) \right] \tag{12}$$

- (c) For Bernoulli-distributed likelihood function, we define:

$$P_{Bernoulli}(y|x, W) = \prod_{m=1}^M P_{Bernoulli}(y^m|x, W) \tag{13}$$

$$P_{Bernoulli}(y^m = 1|x, W) = \mu^m \tag{14}$$

$$P_{Bernoulli}(y^m|x, W) = (\mu^m)^{y^m} (1 - \mu^m)^{1-y^m} \tag{15}$$

where the denotations are the same as those in Eqs. (7) and (8).

By taking logarithm and negative operations on Eq. (13), the loss function $L_{Bernoulli}(W|x_n, y_n)$ for the Bernoulli-distributed

likelihood function given the training data pair (x_n, y_n) is:

$$L_{Bernoulli}(W|x_n, y_n) = \sum_{m=1}^M [(y_n^m - 1)\log(1 - \mu_n^m) - y_n^m \log(\mu_n^m)] \tag{16}$$

We would like to learn the weights to maximize Eqs. (7), (10) and (13) in the training dataset, which is equivalent to minimizing the loss functions defined in Eqs. (9), (12) and (16). There are two channels (μ and σ) in the BCNN output for Laplacian-distributed and Gaussian-distributed likelihood functions while there is only one channel (μ) in the BCNN output for the Bernoulli-distributed likelihood function.

To measure the model uncertainty, we use the dropout network³³. A distribution $q(W)$ is learned to approximate $p(W|X, Y)$ (minimizing the Kullback-Leibler divergence between $q(W)$ and $p(W|X, Y)$) by applying a dropout layer before every layer that has learnable weights. During the prediction process, the model uncertainty is approximated by Monte Carlo dropout³³. With Monte Carlo integration, the predictive distribution $p(y|x_{test,t}, X, Y)$ in Eq. (6) can be approximated as:

$$p(y|x_{test,t}, X, Y) \approx \int p(y|x_{test,t}, W)q(W)dW \approx \frac{1}{K} \sum_{k=1}^K p(y|x_{test,t}, W^k) \tag{17}$$

where K is the total number of dropout activations during the prediction process.

Finally, the predicted image can be represented by the predicted mean $\hat{\mu}_{test,t}^m$ of the m th pixel for the testing data $x_{test,t}$ (for Laplacian-distributed, Gaussian-distributed and Bernoulli-distributed likelihood functions) is:

$$\hat{\mu}_{test,t}^m = \mathbb{E}[y^m|x_{test,t}, X, Y] \approx \frac{1}{K} \sum_{k=1}^K \mathbb{E}[y^m|x_{test,t}, W^k] \approx \frac{1}{K} \sum_{k=1}^K \hat{\mu}_{test,t}^{m,k} \tag{18}$$

where \mathbb{E} denotes the expectation and $\hat{\mu}_{test,t}^{m,k}$ denotes the predicted μ of the m th pixel and k th dropout activation for the testing data $x_{test,t}$.

The predicted uncertainty $\hat{\sigma}_{test,t}^m$ of the m th pixel for the testing data $x_{test,t}$ for Laplacian-distributed likelihood function is:

$$\begin{aligned} \hat{\sigma}_{test,t}^m(Laplacian) &= \sqrt{\text{Var}(y^m|x_{test,t}, X, Y)} \\ &= \sqrt{\mathbb{E}[\text{Var}(y^m|x_{test,t}, W, X, Y)] + \text{Var}(\mathbb{E}[y^m|x_{test,t}, W, X, Y])} \\ &= \sqrt{\mathbb{E}[\text{Var}(y^m|x_{test,t}, W)] + \text{Var}(\mathbb{E}[y^m|x_{test,t}, W])} \\ &\approx \sqrt{\frac{1}{K} \sum_{k=1}^K 2(\hat{\sigma}_{test,t}^{m,k})^2 + \frac{1}{K} \sum_{k=1}^K (\hat{\mu}_{test,t}^{m,k} - \hat{\mu}_{test,t}^m)^2} = \sqrt{(\hat{\sigma}_{test,t}^{m(D)})^2 + (\hat{\sigma}_{test,t}^{m(M)})^2} \end{aligned} \tag{19}$$

where Var denotes pixel-wise variance, $\hat{\sigma}_{test,t}^{m,k}$ denotes the predicted standard deviation of the m th pixel and k th dropout activation for the testing data $x_{test,t}$. $\hat{\sigma}_{test,t}^{m(D)} = \sqrt{\frac{1}{K} \sum_{k=1}^K 2(\hat{\sigma}_{test,t}^{m,k})^2}$ denotes the data uncertainty and $\hat{\sigma}_{test,t}^{m(M)} = \sqrt{\frac{1}{K} \sum_{k=1}^K (\hat{\mu}_{test,t}^{m,k} - \hat{\mu}_{test,t}^m)^2}$ denotes the model uncertainty.

For Gaussian-distributed likelihood function:

$$\begin{aligned} \hat{\sigma}_{test,t}^m(Gaussian) &\approx \sqrt{\frac{1}{K} \sum_{k=1}^K (\hat{\sigma}_{test,t}^{m,k})^2 + \frac{1}{K} \sum_{k=1}^K (\hat{\mu}_{test,t}^{m,k} - \hat{\mu}_{test,t}^m)^2} \\ &= \sqrt{(\hat{\sigma}_{test,t}^{m(D)})^2 + (\hat{\sigma}_{test,t}^{m(M)})^2} \end{aligned} \tag{20}$$

where the denotations are the same as those in Eq. (19) and the derivation of Eq. (20) is similar to that of Eq. (19).

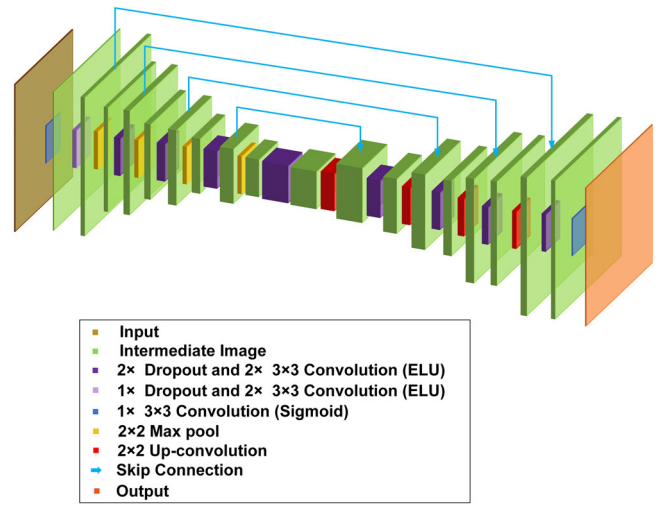


Fig. 6 The BCNN structure. The BCNN uses a U-Net architecture with an encoder-decoder structure. Each level of the U-Net includes dropout and convolutional (with ELU or Sigmoid activations) layers. The encoder and decoder are connected through skip connections.

For Bernoulli-distributed likelihood function:

$$\begin{aligned} \hat{\sigma}_{test,t}^m(Bernoulli) &\approx \sqrt{\frac{1}{K} \sum_{k=1}^K [\hat{\mu}_{test,t}^{m,k}(1 - \hat{\mu}_{test,t}^{m,k})] + \frac{1}{K} \sum_{k=1}^K (\hat{\mu}_{test,t}^{m,k} - \hat{\mu}_{test,t}^m)^2} \\ &= \sqrt{(\hat{\sigma}_{test,t}^{m(D)})^2 + (\hat{\sigma}_{test,t}^{m(M)})^2} \end{aligned} \tag{21}$$

where the denotations are the same as those in Eq. (19) and the derivation of Eq. (21) is similar to that of Eq. (19).

We can find from Eqs. (19–21) that the data uncertainty ($\hat{\sigma}_{test,t}^{m(D)}$) is approximated by the mean of the predicted variance and the model uncertainty ($\hat{\sigma}_{test,t}^{m(M)}$) is approximated by the variance of the predicted mean.

BCNN structures. The BCNN structures are shown in Fig. 6. They follow the U-Net architecture⁴⁰, which utilizes an encoder-decoder structure with skip connections to preserve wide-frequency features. This architecture was chosen because of its success in solving image-to-image problems. Dropout layers with a dropout rate of 0.1 were included before each convolution layer of the U-Net in order to prevent overfitting during the training process. L₂ kernel regularizer and bias regularizer with the regularization factor of 1×10^{-6} were included in each convolution layer. The network structure in Fig. 6 is used for Bernoulli-distributed likelihood function. For Laplacian-distributed and Gaussian-distributed likelihood functions, the same architecture is used except that there are two output channels (for μ and σ). The loss functions in Eqs. (9), (12) and (16) were used in BCNN for Laplacian-distributed, Gaussian-distributed and Bernoulli-distributed likelihood function, respectively. The BCNN was trained on a NVIDIA Quadro M4000 GPU with an 8GB of memory.

Data simulation and pre-processing. RD Hadamard⁴¹ patterns are used as the sampling patterns in the simulated SPI. In RD Hadamard patterns, the measurement order of the Hadamard basis is reordered and optimized according to their significance for general scenes, such that at discretized increments, the complete sampling for different spatial frequencies is obtained⁴¹.

The MNIST database³⁶ was used for training the BCNN with 800 images as the training dataset, 100 images as the validating

dataset and another 100 images as the testing dataset. All the images were normalized, converted to binary images and up-sampled from 28×28 to 32×32 to meet the dimension requirement of the RD Hadamard patterns. The full RD Hadamard basis for a 32×32 image has 1024 RD Hadamard patterns each with a size of 32×32 . Varying compression ratios (varying levels of model ill-posedness) were used here as $8 \times$, $16 \times$, $32 \times$ and $64 \times$ corresponding to taking the first $1/8$, $1/16$, $1/32$ and $1/64$ of the RD Hadamard patterns, respectively. The 1D raw measurement data were acquired by multiplying each individual image with the RD Hadamard patterns at each compression ratio. Therefore, the 1D raw measurement data have a size of 128×1 , 64×1 , 32×1 and 16×1 for the corresponding compression ratios. Finally, white Gaussian noise was added to the 1D measurement data to achieve an SNR of 25 dB. The SNR is defined as:

$$\text{SNR} = 20 \log_{10} \frac{\text{averaged signal amplitude}}{\text{standard deviation of noise}} \quad (22)$$

The STL-10 natural image database³⁷ was used for training the BCNN with 10,000 images as the training dataset, 2000 images as the validating dataset and another 2000 images as the testing dataset. All the images were down-sampled from 96×96 to 64×64 to meet the dimension requirement of the RD Hadamard patterns. The full RD Hadamard basis for a 64×64 image has 4096 RD Hadamard patterns each with a size of 64×64 . Varying compression ratios (varying levels of model ill-posedness) were used here as $2 \times$, $4 \times$, $8 \times$ and $16 \times$. Therefore, the 1D raw measurement data have a size of 2048×1 , 1024×1 , 512×1 and 256×1 for the corresponding compression ratios. Finally, white Gaussian noise was added to the 1D measurement data to achieve an SNR of 25 dB.

In DL, a pre-processing step is usually used to convert the inputs of the neural network from measurement domain into image domain and therefore makes the learning process easier^{21,24,42,43}. In this paper, the pre-processing step is a linear operation on the acquired raw SPI data to reconstruct an initial guess of each image in the training, validating and testing datasets using the approximant inverse model matrix, and then used as the input of BCNN for further training and prediction.

In order to efficiently compute the pseudoinverse of the large forward model matrix, \mathbf{H} , a computational approach was employed. The equation, $\mathbf{H}\mathbf{H}_{\text{inv}} = \mathbf{I}$ was solved one column at a time, where \mathbf{H}_{inv} is the pseudoinverse of \mathbf{H} and \mathbf{I} is the identity matrix. Thus, to calculate the i^{th} column of \mathbf{H}_{inv} , the LSQR method in Matlab was applied using \mathbf{H} and the i^{th} column of \mathbf{I} ³⁵. In the case of the simulations, which relied on the Russian Doll Hadamard matrix, the result was equivalent to the transpose of \mathbf{H} .

For the BCNN trained with the MNIST database, the Adam optimizer was used with a linearly decreasing learning rate starting from 5×10^{-4} and ending with 5×10^{-6} . The batch size was chosen to be 40 and the BCNN was trained for 500 epochs to guarantee a complete training. The overall training time was approximately 7 minutes. For the BCNN trained with the STL-10 database, the Adam optimizer was used with a constant learning rate of 5×10^{-4} . The batch size was chosen to be 50 and the BCNN was trained for 200 epochs to guarantee a complete training. The overall training time was approximately 70 min.

Experimental data acquisition and pre-processing. Random grayscale illumination patterns were used in the experimental SPI. The images were taken from MNIST database³⁶, normalized, converted to binary images and resized from 28×28 to 32×32 pixels. 1024 random grayscale illumination patterns each with a size of 32×32 were prepared as the full measurement basis. Then,

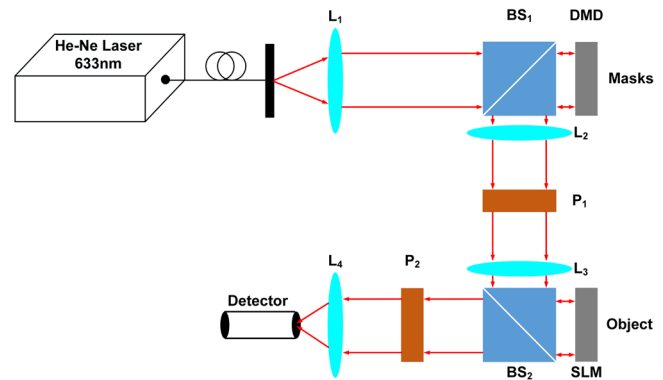


Fig. 7 The imaging system. L_1 , L_2 , L_3 and L_4 are optical lenses whose focal lengths are 80 mm, 80 mm, 80 mm and 40 mm respectively. P_1 is a horizontally polarized linear polarizer, and P_2 is a vertically polarized linear polarizer. BS_1 and BS_2 are beam splitters. DMD is a digital micro-mirror device to display mask patterns. SLM is a spatial light modulator to display the object patterns. An sCMOS camera is used as the bucket detector by integrating all the pixels of each acquired image to produce the single-pixel signal.

the first 64 or 16 illumination patterns in the full basis were used to illuminate the objects, corresponding to a $16 \times$ or $64 \times$ compression ratio, respectively. Therefore, the corresponding 1D raw measurement data have a size of 64×1 or 16×1 . The imaging system is shown in Fig. 7. A spatial light modulator (Pluto-Vis, Holoeye Photonics AG) is used to display each image and then the image is illuminated by a set of random grayscale sampling patterns which are displayed on a digital micromirror device²⁶. The 1D measurement data were collected by a bucket detector. An sCMOS camera (Zyla 4.2 PLUS sCMOS, Andor Technology Ltd) was used as the bucket detector by integrating all the pixels of each acquired image to produce the single-pixel signal²⁶. In the pre-processing step, an initial guess of each image in the dataset is reconstructed using the LSQR-approximated inverse model matrix, and then used as the input of BCNN for further training and prediction. The BCNN was trained on an experimentally acquired dataset of 800 images and tested on 100 images with the Bernoulli-distributed likelihood function. The batch size was chosen to be 40 and the BCNN was trained for 500 epochs to guarantee a complete training. The overall training time was approximately 7 min.

Data availability

The data to implement the BCNN in simulated $16 \times$ SPI with MNIST dataset is available at <https://github.com/FMILab/Single-Pixel-Imaging-with-Uncertainty-Approximation>. Other generated and/or analyzed datasets that support the findings of this study are available from the corresponding author upon reasonable request.

Code availability

The code to implement and analyze the BCNN is available at <https://github.com/FMILab/Single-Pixel-Imaging-with-Uncertainty-Approximation>.

Received: 17 January 2023; Accepted: 23 July 2023;

Published online: 01 August 2023

References

- Gibson, G. M., Johnson, S. D. & Padgett, M. J. Single-pixel imaging 12 years on: a review. *Opt. Express* **28**, 28190–28208 (2020).
- Candès, E. J. & Wakin, M. B. An introduction to compressive sampling. *IEEE Signal Process. Mag.* **25**, 21–30 (2008).

3. Pittman, T. B., Shih, Y., Strekalov, D. & Sergienko, A. V. Optical imaging by means of two-photon quantum entanglement. *Phys. Rev. A* **52**, R3429 (1995).
4. Hoshi, I., Shimobaba, T., Kakue, T. & Ito, T. Single-pixel imaging using a recurrent neural network combined with convolutional layers. *Opt. Express* **28**, 34069–34078 (2020).
5. Erkmén, B. I. Computational ghost imaging for remote sensing. *JOSA A* **29**, 782–789 (2012).
6. Clemente, P. et al. Compressive holography with a single-pixel detector. *Opt. Lett.* **38**, 2524–2527 (2013).
7. Endo, Y., Tahara, T. & Okamoto, R. Color single-pixel digital holography with a phase-encoded reference wave. *Appl. Opt.* **58**, G149–G154 (2019).
8. Zhang, Z., Jiao, S., Yao, M., Li, X. & Zhong, J. Secured single-pixel broadcast imaging. *Opt. Express* **26**, 14578–14591 (2018).
9. Jiao, S., Zhou, C., Shi, Y., Zou, W. & Li, X. Review on optical image hiding and watermarking techniques. *Opt. Laser Technol.* **109**, 370–380 (2019).
10. Peng, J. et al. Micro-tomography via single-pixel imaging. *Opt. Express* **26**, 31094–31105 (2018).
11. Candes, E. & Romberg, J. Sparsity and incoherence in compressive sampling. *Inverse Probl.* **23**, 969 (2007).
12. Candès, E. J. & Wakin, M. B. An introduction to compressive sampling [a sensing/sampling paradigm that goes against the common knowledge in data acquisition]. *IEEE Signal Process. Mag.* **25**, 21–30 (2008).
13. Shang, R., Hoffer-Hawlik, K., Wang, F., Situ, G. & Luke, G. P. Two-step training deep learning framework for computational imaging without physics priors. *Opt. Express* **29**, 15239–15254 (2021).
14. Barbastathis, G., Ozcan, A. & Situ, G. On the use of deep learning for computational imaging. *Optica* **6**, 921–943 (2019).
15. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436 (2015).
16. Rivenson, Y. et al. Deep learning microscopy. *Optica* **4**, 1437–1443 (2017).
17. Rivenson, Y., Zhang, Y., Günaydin, H., Teng, D. & Ozcan, A. Phase recovery and holographic image reconstruction using deep learning in neural networks. *Light: Sci. Appl.* **7**, 17141 (2018).
18. Li, Y., Xue, Y. & Tian, L. Deep speckle correlation: a deep learning approach toward scalable imaging through scattering media. *Optica* **5**, 1181–1190 (2018).
19. Xue, Y., Cheng, S., Li, Y. & Tian, L. Reliable deep-learning-based phase imaging with uncertainty quantification. *Optica* **6**, 618–629 (2019).
20. Sinha, A., Lee, J., Li, S. & Barbastathis, G. Lensless computational imaging through deep learning. *Optica* **4**, 1117–1125 (2017).
21. Goy, A., Arthur, K., Li, S. & Barbastathis, G. Low photon count phase retrieval using deep learning. *Phys. Rev. Lett.* **121**, 243902 (2018).
22. Goy, A. et al. High-resolution limited-angle phase tomography of dense layered objects using deep neural networks. *Proc. Natl. Acad. Sci. USA.* **116**, 19848–19856 (2019).
23. Higham, C. F., Murray-Smith, R., Padgett, M. J. & Edgar, M. P. Deep learning for real-time single-pixel video. *Sci. Rep.* **8**, 1–9 (2018).
24. Lyu, M. et al. Deep-learning-based ghost imaging. *Sci. Rep.* **7**, 1–6 (2017).
25. Shimobaba, T. et al. Computational ghost imaging using deep learning. *Opt. Commun.* **413**, 147–151 (2018).
26. Wang, F., Wang, H., Wang, H., Li, G. & Situ, G. Learning from simulation: An end-to-end deep-learning approach for computational ghost imaging. *Opt. Express* **27**, 25560–25572 (2019).
27. Jiao, S. et al. Optical machine learning with incoherent light and a single-pixel detector. *Opt. Lett.* **44**, 5186–5189 (2019).
28. Wang, Z., Bovik, A. C., Sheikh, H. R. & Simoncelli, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* **13**, 600–612 (2004).
29. Kendall, A. & Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? *Adv. Neural Inf. Process. Syst.* **30**, 5574–5584 (2017).
30. Feng, S., Zuo, C., Hu, Y., Li, Y. & Chen, Q. Deep-learning-based fringe-pattern analysis with uncertainty estimation. *Optica* **8**, 1507–1510 (2021).
31. Kwon, Y., Won, J.-H., Kim, B. J. & Paik, M. C. Uncertainty quantification using Bayesian neural networks in classification: Application to biomedical image segmentation. *Comput. Stat. Data Anal.* **142**, 106816 (2020).
32. Jospin, L. V., Laga, H., Boussaid, F., Buntine, W. & Bennamoun, M. Hands-on Bayesian neural networks—A tutorial for deep learning users. *IEEE Comp. Intell. Mag.* **17**, 29–48 (2022).
33. Gal, Y. & Ghahramani, Z. In *International Conference on Machine Learning*. 1050–1059 (PMLR).
34. Lakshminarayanan, B., Pritzel, A. & Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *dv. Neural Inf. Process. Syst.* **30** (2017).
35. Paige, C. C. & Saunders, M. A. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw. (TOMS)* **8**, 43–71 (1982).
36. LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **86**, 2278–2324 (1998).
37. Coates, A., Ng, A. & Lee, H. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 215–223 (JMLR Workshop and Conference Proceedings).
38. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
39. Hanin, B. & Rolnick, D. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 569–579.
40. Ronneberger, O., Fischer, P. & Brox, T. In *International Conference on Medical image computing and computer-assisted intervention*. 234–241 (Springer).
41. Sun, M.-J., Meng, L.-T., Edgar, M. P., Padgett, M. J. & Radwell, N. A Russian Dolls ordering of the Hadamard basis for compressive single-pixel imaging. *Sci. Rep.* **7**, 3464 (2017).
42. Kim, M., Jeng, G.-S., Pelivanov, I. & O'Donnell, M. Deep-learning image reconstruction for real-time photoacoustic system. *IEEE Trans. Med. Imaging* **39**, 3379–3390 (2020).
43. Wang, F., Wang, C., Deng, C., Han, S. & Situ, G. Single-pixel imaging using physics enhanced deep learning. *Photonics Res.* **10**, 104–110 (2022).

Acknowledgements

The authors acknowledge fundings from NIH grant R21GM137334, a Faculty Grant from the Neukom Institute for Computational Science at Dartmouth College and the Alma Hass Milham Fellowship in Biomedical Engineering from Thayer School of Engineering at Dartmouth College.

Author contributions

R.S. and G.P.L. conceived the idea. R.S. wrote the code, simulated the data, and analyzed the results. M.A.O. optimized the parameters in the code. F.W. and G.S. conducted the experiment. R.S. and G.P.L. wrote the manuscript. F.W. and G.S. revised the manuscript.

Competing interests

The authors declare no competing interests.

Additional information


Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s44172-023-00103-1>.

Correspondence and requests for materials should be addressed to Geoffrey P. Luke.

Peer review information *Communications Engineering* thanks the anonymous reviewers for their contribution to the peer review of this work. Primary Handling Editors: Rosamund Daw, Mengying Su.

Reprints and permission information is available at <http://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) 2023