



Random two-frame interferometry based on deep learning

ZIQIANG LI,^{1,2,3,4}  XINYANG LI,^{1,2} AND RONGGUANG LIANG^{4,*}

¹Key Laboratory on Adaptive Optics, Chinese Academy of Sciences, Chengdu, Sichuan 610209, China

²Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu, Sichuan 610209, China

³University of Chinese Academy of Sciences, Beijing 100049, China

⁴College of Optical Sciences, University of Arizona, Tucson, Arizona 85721, USA

*rliang@optics.arizona.edu

Abstract: A two-frame phase-shifting interferometric wavefront reconstruction method based on deep learning is proposed. By learning from a large number of simulation data based on a physical model, the wrapped phase can be calculated accurately from two interferograms with an unknown phase step. The phase step can be any value excluding the integral multiples of π and the size of interferograms can be flexible. This method does not need a pre-filtering to subtract the direct-current term, but only needs a simple normalization. Comparing with other two-frame methods in both simulations and experiments, the proposed method can achieve better performance.

© 2020 Optical Society of America under the terms of the [OSA Open Access Publishing Agreement](#)

1. Introduction

As one of the most popular techniques in optical metrology [1], traditional Phase-shifting interferometry (PSI) requires three or more interferograms with fixed, known phase steps [2–4]. However, the acquisition of these interferograms is time-consuming and sensitive to mechanical vibrations, ambient air turbulence and temperature changes. Thus, the number of interferograms should be reduced to minimize the recording time. But it is difficult to reconstruct the phase with only one interferogram. For example, only one interferogram cannot distinguish the positive or negative of defocus, therefore, it is difficult to separate convex surfaces from concave surfaces with one interferogram. If there is no additional prior information, at least two frames of interferograms are needed to solve the sign ambiguity problem. Takeda *et al.* developed a Fourier-transform method to extract the phase from a single interferogram [5]. By introducing a large spatial-carrier frequency, the phase information can be separated from unwanted irradiance variations in Fourier domain. However, the limitation is that an interferogram with closed-fringes cannot be applied. Thus, a large tilt is needed to generate the wanted spatial carrier, which will make the fringes denser. If the fringes are too dense, not only the camera cannot record them, but also it will cause considerable retrace error [6,7]. To simplify the measurement process and reduce the instrument cost, the phase reconstruction using two frames of interferograms has been investigated extensively in the past decade [8–11]. In reference [8], a demodulation method, called Kreis method, based on the Fourier transform of two interferograms was proposed. It can demodulate the phase from two interferograms without sign ambiguity. It is one of the most classical algorithms in this field, but it is very sensitive to noise. In [9], a two-step interferometric method based on a regularized optical flow algorithm (OF) was proposed. This method does not need to normalize the fringe pattern, but needs to subtract the direct-current (DC) term. In reference [10], a phase reconstruction method based on Gram Schmidt (GS) orthogonalization with two fringe patterns as independent vectors was proposed. Because of its high accuracy and small computation consumption, this method has achieved great success and is widely used in two-frame interferometry. This method also requires the DC term be removed. In reference [11], a fast and accurate (FA) two-frame PSI wavefront reconstruction method was proposed. Before

phase reconstruction, it is also necessary to use a high-pass Gaussian filter to filter out the DC term, and then estimate the cosine value of the unknown phase step between two interferograms by solving a quartic polynomial equation, and finally calculate the wrapped phase.

In recent years, deep learning (DL) technology based on artificial neural networks (ANN) has developed rapidly, especially in the field of computer vision. In reference [12], a convolutional neural network named U-Net was proposed for biomedical image segmentation. In reference [13], the U-Net was modified to be the generator in a Generative Adversarial Networks (GANs) applying to Image-to-Image Translation. These latest developments in computer vision have inspired the field of optical metrology, such as phase unwrapping and denoising [14–16]. In reference [17], Kando *et al.* proposed a deep learning-based method to extract phases from single-shot interferograms, even when the interferogram includes closed ring-shaped fringes. However, this method is not suitable for the precise measurement of freeform surfaces because it cannot be applicable to interferograms including more than one closed-fringes. In order to solve the shortcomings of the above methods, we modified the generator in [13], and proposed a new two-frame phase-shifting method based on deep learning. We call the proposed network the Phase U-Net (PUN). The proposed PUN neither needs a spatial carrier, nor a filter for subtracting the DC term, but only a simple normalization. It can accurately recover the wrapped phase from two interferograms with an unknown phase step excluding the singular case, which corresponds to integral multiples of π .

The paper is organized in the following way: In Section 2 we analyze the advantages of two-frame methods over one-frame methods. Then we illustrate how we generated the datasets for training the network and explain the architecture of the PUN, including details in the training process in Section 3. In Section 4, we do simulations and experiments to compare the algorithms and make a detailed error analysis, followed by the conclusion.

2. Advantages of two-frame methods over one-frame methods

In traditional four-step PSI [18], the four phase-shifted interferograms can be described as:

$$I_1(x, y) = a(x, y) + b(x, y) \cos[W(x, y)], \quad (1)$$

$$I_2(x, y) = a(x, y) + b(x, y) \cos \left[W(x, y) + \frac{1}{2}\pi \right], \quad (2)$$

$$I_3(x, y) = a(x, y) + b(x, y) \cos[W(x, y) + \pi], \quad (3)$$

$$I_4(x, y) = a(x, y) + b(x, y) \cos \left[W(x, y) + \frac{3}{2}\pi \right]. \quad (4)$$

where I_1 , I_2 , I_3 and I_4 are the intensities of the four interferograms respectively, a means the DC term, b means the modulation term, W is the original phase, and (x, y) is the coordinate.

We can filter out the unwanted background intensities a and b by using the equation below:

$$\phi = \arctan \left(\frac{I_4 - I_2}{I_1 - I_3} \right). \quad (5)$$

where ϕ is the wrapped phase of the original phase W .

The traditional four-step PSI requires four interferograms with accurate phase steps of $\pi/2$. This method is sensitive to mechanical vibrations, ambient air turbulence and temperature changes. Thus, many advanced algorithms have been proposed to reduce the number of interferograms.

The Fourier-transform method was first proposed by Takeda *et al.* [5]. If the tilt is not set to zero, the fringe pattern can be expressed as:

$$I(x, y) = a(x, y) + b(x, y) \cos[2\pi f_0 x + W(x, y)]. \quad (6)$$

where f_0 is called the spatial-carrier frequency introduced by the tilt. In most cases a , b and W vary slowly compared with f_0 .

An image-sensing device (such as CCD and CMOS) that has enough resolution to satisfy the sampling-theory requirement is used to capture the fringes. The fringe pattern is rewritten in the following form for convenience of explanation:

$$I(x, y) = a(x, y) + c(x, y)\exp(2\pi if_0x) + c^*(x, y)\exp(-2\pi if_0x), \quad (7)$$

with

$$c(x, y) = \frac{1}{2}b(x, y)\exp[iW(x, y)]. \quad (8)$$

where $*$ denotes a complex conjugate.

Then the fast-Fourier-transform (FFT) algorithm is used to transform Eq. (7) into Fourier domain.

$$FI(fx, fy) = A(fx, fy) + C(fx - f_0, fy) + C^*(fx + f_0, fy). \quad (9)$$

where FI means the Fourier spectra of intensity I , and the other capital letters denote the corresponding Fourier spectra in Eq. (7). (fx, fy) are the spatial frequency in the x direction and y direction respectively. Since the spatial variations of a , b and W are slow compared with the spatial frequency f_0 , the Fourier spectra in Eq. (9) are separated by the carrier frequency f_0 . Thus, $C(fx - f_0, fy)$ can be extracted and to obtain $C(fx, fy)$. Note that the unwanted background variation a has been filtered out in this stage.

Again, using the FFT algorithm, we compute the inverse Fourier transform of $C(fx, fy)$ with respect to (fx, fy) and obtain $c(x, y)$, defined by Eq. (8). Then we calculate a complex logarithm of Eq. (8):

$$\log[c(x, y)] = \log\left[\frac{1}{2}b(x, y)\right] + i\phi(x, y). \quad (10)$$

Now we have the wrapped phase $\phi(x, y)$ in the imaginary part completely separated from the unwanted amplitude variation $b(x, y)$ in the real part. However, if the spatial-carrier frequency is not large enough, this method becomes invalid. Besides, this kind of cross-talk in Fourier domain corresponds to closed-fringes in spatial domain. In other words, there is a limitation of the Fourier-transform method, that an interferogram including closed-fringes cannot be applied.

Each closed-fringe pattern in interferograms corresponds to two possible situations: a concave surface or a convex surface. This is a typical one-to-multiple mapping relationship. In fact, a one-to-one mapping relationship can be determined by appointing all closed-fringe patterns to be concave surfaces (or convex surfaces) [19]. Based on this assumption, a deep learning-based method or an improved Fourier-transform method can be proposed to extract phases from single-shot interferograms with closed-fringe patterns [17,19]. However, freeform surfaces can be considered to be composed of many convex surfaces and concave surfaces, we can never really know whether they are convex or concave in only one interferogram [19]. For example, Fig. 1(a) shows a sphere with two small defects in the center: one of them can be seen as a small convex surface while the other one can be seen as a small concave surface. A tilt has been added to the sphere but is not large enough to turn closed-fringes in the interferogram into open-fringes. Figure 1(b) is the interferogram of that sphere. There are two points P1 and P2 on the adjacent fringes as shown in Fig. 1(b). It is known that the phase difference between P1 and P2 is 2π but we cannot know which point is higher, and it is the same with the other defect. Thus, there are four possibilities for those two small defects with the same interferogram in Fig. 1(b), and the other three possible situations are shown in Fig. 2.

Two-frame methods do not have the sign ambiguity problem. Figure 3(a) shows the second interferograms for those four surface shapes in Fig. 1(a) and Fig. 2. Those interferograms are different so that the four situations above can be well distinguished when combined with the first interferograms in Fig. 1(b). Figure 3(b) shows the wrapped phases calculated by the PUN and Fig. 3(c) shows the unwrapped phases unwrapped from Fig. 3(b). As we can see, the PUN can reconstruct those four different surfaces correctly.

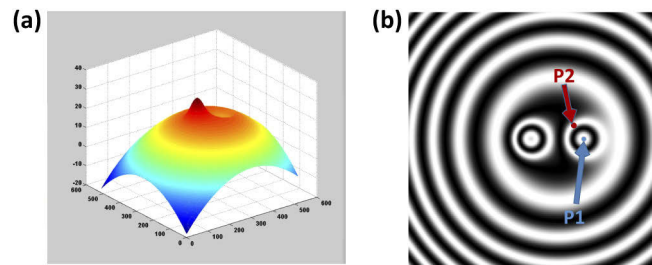


Fig. 1. (a) A sphere with two little defects and a small tilt. (b) The interferogram of (a).

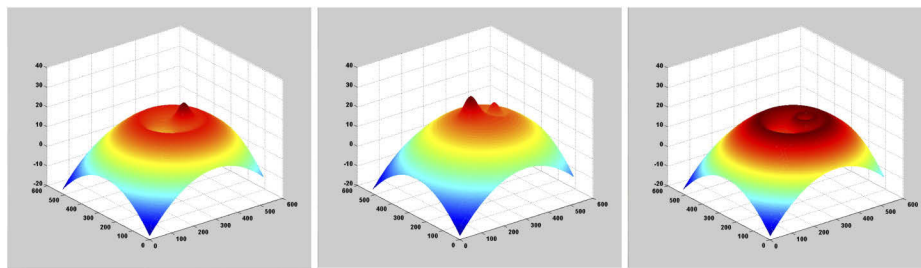


Fig. 2. The other three possible situations corresponding to the same interferogram in Fig. 1(b).

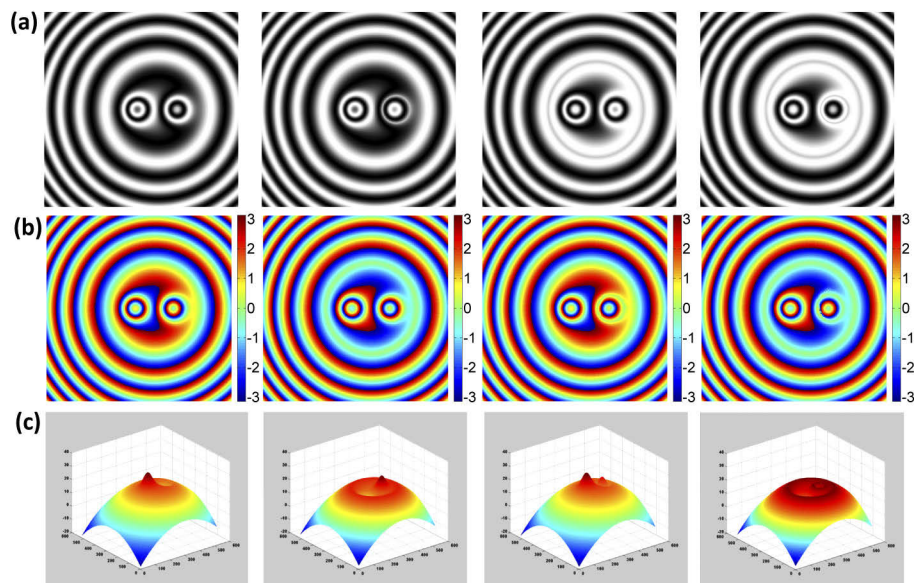


Fig. 3. (a) Interferograms for the four different surface shapes in Fig. 1(a) and Fig. 2. (b) The wrapped phases calculated by the PUN using two interferograms. (c) The unwrapped phases.

In conclusion, whether the Fourier-transform method or the deep learning-based one-frame method, in order to fully solve the sign ambiguity problem, all kinds of one-frame interferometry methods need a large tilt which may cause too dense fringe patterns and the retrace error, while two-frame methods do not need such tilts and are more suitable for high precision optical metrology. Therefore, in Section 4, we will compare the PUN with other two-frame methods rather than one-frame methods.

3. Data generation and the training process

3.1. Data generation

All deep learning-based methods start with data. In this paper, we generate the data by simulation for training. In order to enhance the generalization ability of the network and make it perform well for all possible situations, we use two different ways to generate the original phase. The first way is to use the Zernike polynomials. Since any continuous wavefront shape may be represented by a linear combination of the Zernike polynomials [18], the original phase W can be expressed as follows.

$$W(\rho, \theta) = \sum_{r=0}^L Z_r U_r(\rho, \theta), \quad (11)$$

where L is the maximum power, Z_r is the Zernike coefficient, ρ and θ are radius and angle respectively. The single index r is defined by:

$$r = \frac{n(n+1)}{2} + m + 1, \quad (12)$$

and the polynomial U_r can be expressed as follows.

$$U_n^{n-2m}(\rho, \theta) = R_n^{n-2m}(\rho) \begin{cases} \sin \\ \cos \end{cases} (n-2m)\theta, \quad (13)$$

where the sine function is used when $n-2m>0$, and the cosine function is used when $n-2m \leq 0$. The radial polynomial is given by:

$$R_n^{n-2m}(\rho) = \sum_{s=0}^m (-1)^s \frac{(n-s)!}{s!(m-s)!(n-m-s)!} \rho^{n-2s}. \quad (14)$$

The other way is to introduce the actuator influence function which is commonly used in adaptive optics to model the surface of deformable mirrors [20].

$$P_i(x, y) = \exp \left\{ \ln(k) \left[\frac{\sqrt{(x-u_i)^2 + (y-v_i)^2}}{d} \right]^g \right\}. \quad (15)$$

where P_i is the actuator influence function, (x, y) is the coordinate of each point on the original phase, (u_i, v_i) is the coordinate of the actuator. k , d and g are coefficients to simulate different types of deformable mirrors and will have little impact to the final results in this paper, because they are still the same type of interferograms even when the coefficients change. k is the coupling coefficient and is set to 0.15 in this paper, d is the actuator spacing and is set to 8, g is the Gaussian index and is set to 2.5. So, the original phase is obtained as follows.

$$W(x, y) = \sum_{i=1}^N H_i P_i(x, y) \quad (16)$$

where W is the original phase, H_i is the influence amplitude coefficient and N means the number of actuators.

We used several groups of Zernike coefficients (up to 11 terms, including the tip and tilt) to generate original phases. The terms and amplitudes of the Zernike coefficients in each group were carefully selected to prevent from generating too dense fringes. For example, there were 5 terms of Zernike coefficients (1st to 5th order) in Group 1, and the amplitudes of the Zernike coefficients were randomly chosen from -15 to 15 . In Group 2, there were 3 Zernike coefficients (8th, 9th and 10th order) with the amplitudes vary between -10 and 10 . In this way, we could generate many different types of aberrations without too dense fringes in the interferograms. As a supplement to the Zernike polynomials, we also generated original phases by using the actuator influence function. The influence amplitude coefficients were randomly chosen from -25 to 25 . After the original phases were generated, we computed the wrapped phases to be the outputs in the training set. The wrapped phase ϕ is the phase angle of the original phase W , and can be calculated by using the MATLAB function ‘angle’:

$$\phi = \text{angle}[\exp(jW)]. \quad (17)$$

Figure 4 shows an example of two original phases generated by the Zernike polynomials method and the actuator influence function method respectively. Their corresponding wrapped phases and interferograms are also shown.

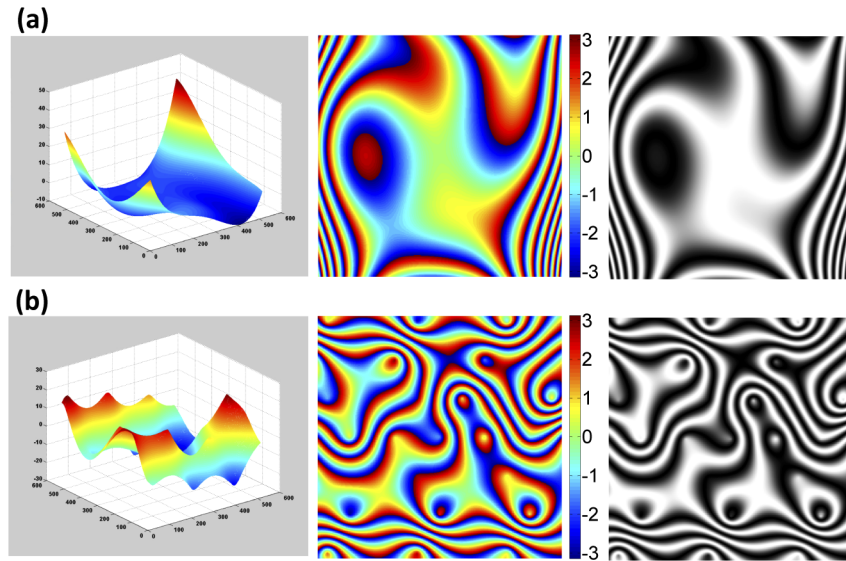


Fig. 4. (a) An original phase generated by the Zernike polynomials method and its corresponding wrapped phase and interferogram. (b) An original phase generated by the influence function method and its corresponding wrapped phase and interferogram.

According to [18], we can model the interferograms in two-frame phase-shifting interferometry as:

$$I_1(x, y) = a(x, y) + b(x, y) \cos[W(x, y)], \quad (18)$$

$$I_2(x, y) = a(x, y) + b(x, y) \cos[W(x, y) + \delta]. \quad (19)$$

where δ is the unknown phase-shifting step.

In order to simulate the non-uniform background intensity and modulation amplitude, a and b were not constants when generating the training set. Also, to better simulate the real situation, additive white Gaussian noise was added into these interferograms and the signal-to-noise ratio (SNR) varied from 20 to 100 dB. Figure 5 shows some simulated interferograms with different

background intensities and noise levels. These interferograms are normalized between 0 and 1 to be the inputs in the training set, while the wrapped phases ϕ are the outputs.

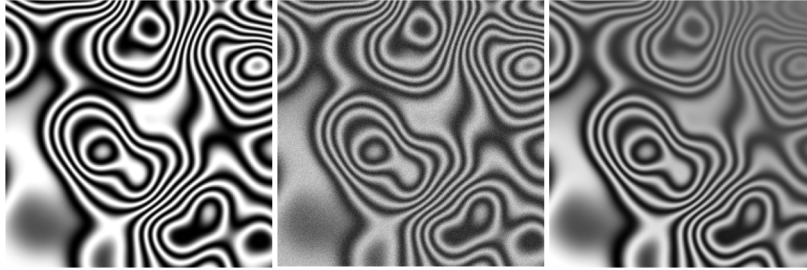


Fig. 5. An example of interferograms with different background intensities and noise levels.

To read the data more efficiently and accelerate the training process, we used the TFRecord format to store the training set. The TFRecord format is a simple format for storing a sequence of binary records. It can be helpful to serialize the data and store it in a set of files (100-200MB each) that can each be read linearly. Thus, we stored 64 pairs of inputs and outputs in one TFRecord format file (192MB). Since the PUN was a deep network, we wanted to generate more than 50,000 pairs of data to train the network. Therefore, we generated 782 TFRecord format files, which were 50,048 pairs of data in total. We also generated a test dataset different from the training set to test the network and to evaluate whether the network was overfitting. Note that the image sizes of both interferograms and wrapped phases were 512×512. The phase step δ should be between 0 and π when generating the training data.

3.2. Architecture of PUN and the training process

The architecture of our network the PUN is illustrated in Fig. 6. Different from the original U-Net in [12], the proposed network has no pooling layer because details are important while pooling layers may lose those features.

In the PUN shown above, we used LeakyReLU [21] activation functions in downsampling steps and ReLU [22] in upsampling steps. Also, Batch Normalization [23] was used in some layers to accelerate training. The Dropout technique [24] was also used in some upsampling steps in order to avoid over-fitting. Different from the original U-Net using softmax in the output layer to solve the image segmentation problem as a classification problem, we used the ELU activation function [25] in the output layer to make predictions of the pixel values. It has little impact that the outputs of the ELU activation function do not range from 0 to 1, because all negative outputs can be easily replaced by zeros and values larger than one can also be replaced by ones in post-processing. We have tried different loss functions and found the mean absolute error loss function performed the best.

$$loss = \frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M (|\phi_{ij} - \chi_{ij}|). \quad (20)$$

where ϕ_{ij} is the (i,j) th value on real wrapped phase ϕ , χ_{ij} is the corresponding value on the predicted wrapped phase, M and N are the image size. We applied the Adam solver [26] to train the PUN on a High Performance Computing cluster using one Intel Xeon E5-2695 V3 CPU, one Nvidia P100 GPU (with 16 GB VRAM), and 224 GB RAM for 120 epochs. The version of Tensorflow framework was 2.0.0.

It is significant to note that there is no fully-connected layer in the network architecture so that we can not only predict wrapped phases using inputs with the size of 512×512, but also larger

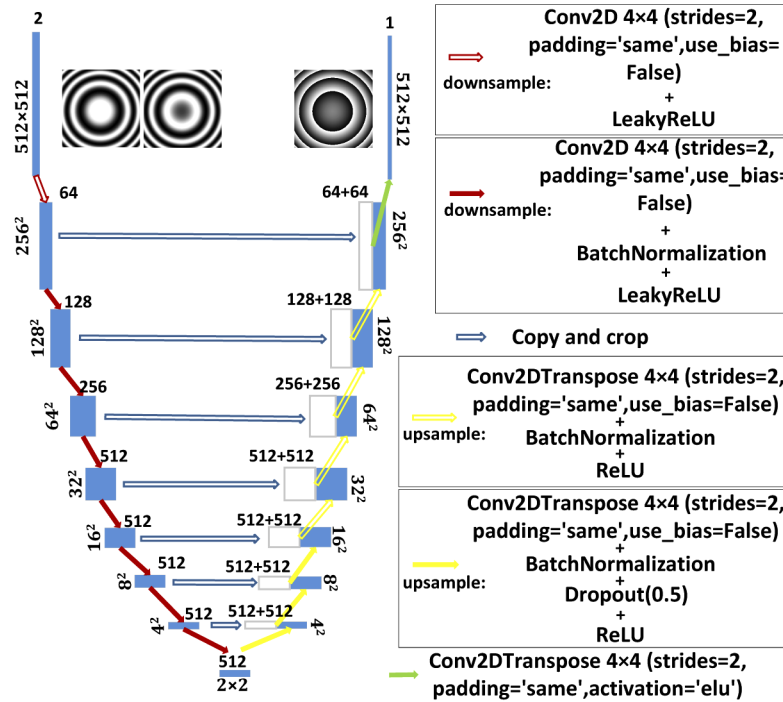


Fig. 6. The architecture of the PUN. Blue boxes are multi-channel feature maps. The number of channels is on the top of the box. The other number is the x-y-size. White boxes represent copied feature maps.

images such as 768×768 , 1024×1024 , and 1280×1280 . However, it doesn't mean that there is no limit in the input size. The input size must be integral multiples of 256×256 . For example, the size of 32×32 is not acceptable. Because the architecture of the PUN consists of two parts: the downsampling part and the upsampling part. In the downsampling part, the image size will be halved from the previous layer of the network to the next layer of the network. The data cannot even reach the bottom layer of the downsampling part when the input size is 32×32 . Most ANNs do not allow users to change the input size for two reasons. One reason is that most of ANNs have fully-connected layers in their network architectures. Changing the input size will lead to the change of the number of parameters in the fully-connected layers. But the number of parameters cannot be changed in a concrete network. Thus, the input sizes of ANNs with fully-connected layers just cannot be changed. The other reason is that the test set (or we can call it the real problem) should be independent and identically distributed (i.i.d.) with the training set. Although in the PUN, there is no fully connected layer and thus it allows users to change the input size, the receptive field of the network is limited by the convolution layers. In the same convolution kernel and with the same original phase, the fringe pattern of a large size interferogram is sparser than the fringe pattern of a small size interferogram. That means, the input size cannot be enlarged without limit because that may break the i.i.d. rule. In one word, this network architecture allows users to choose the input size freely within a certain range. For example, the well-trained PUN in this paper prefers the input sizes between 512×512 to 1280×1280 , and we have tested in Section 4 that the performances fluctuate little in that range. But if the CCD size changed to 2048×2048 , it is better to re-train the network using a new dataset, and that new well-trained network may perform better with input sizes between 2048×2048 to 2816×2816 .

4. Error analysis and experiments

4.1. Error analysis by simulation

We first examined the robustness of the proposed method when phase step changed. We have calculated the RMS errors of the wrapped phases computed by all five two-frame methods at different phase steps among the range from 0 rad to π rad (not including 0 and π). No noise was added to the interferograms and the results are plotted in Fig. 7, showing that the trained PUN is consistently better than other methods. In order to test the robustness under noisy situations, we simulated interferograms with a phase step of 1 rad and added additive white Gaussian noise with SNR ranging from 20 dB to 100 dB and plotted the RMS errors in Fig. 8. This simulation validated that the PUN had the best performance over the whole range of tested SNR.

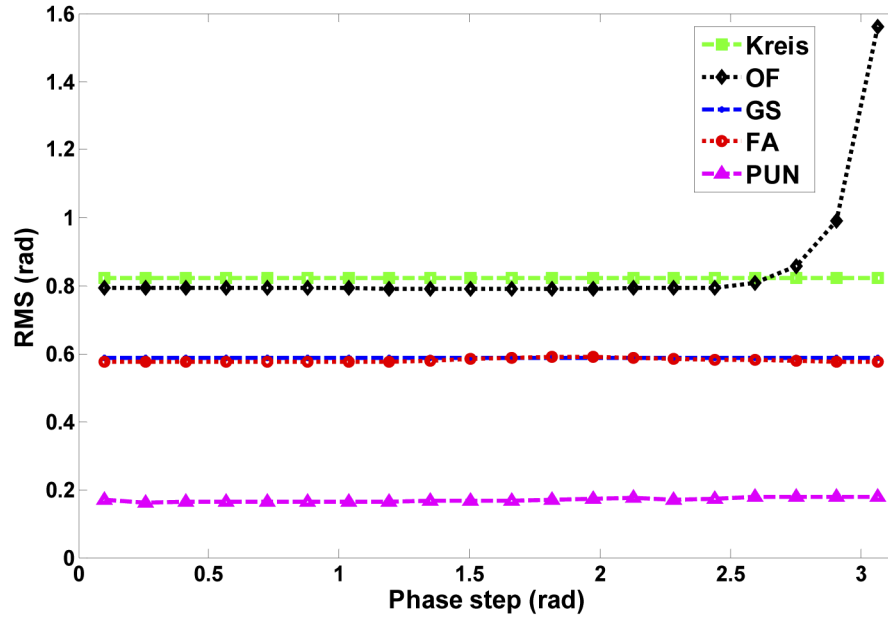


Fig. 7. RMS errors of the Kreis, OF, GS, FA and PUN when phase step changes from 0 to π .

We also tested the PUN with different input sizes to make sure that it would perform well in common image sizes. We generated another two sets of interferograms with sizes of 512×512 , 768×768 , 1024×1024 and 1280×1280 . One set of interferograms had the phase steps of 1 rad and Gaussian noise with 100 dB SNR. The RMS errors of all five methods were calculated in Table 1, followed by the corresponding processing times. We then did the same experiment on the second set which only changed the SNR to 20 dB, as shown in Table 2. It should be noted that the PUN was written in Python and ran in Spyder IDE, while other algorithms were implemented in MATLAB. All methods (including the PUN) ran on a laptop with one Intel Core i5-8250U processor and 8 GB RAM when testing the performances. As can be seen in Table 1 and Table 2, the proposed method PUN is the most accurate among all those methods, while the FA algorithm remains the fastest especially with large image sizes. To have a more intuitive understanding, the results of PUN in Table 1 are demonstrated in Fig. 9.

4.2. Experimental validation

In order to further verify the effectiveness of the proposed method, we collected two sets of interferograms in experiments with $\pi/2$ phase step as shown in Fig. 10(a) and Fig. 12(a), and

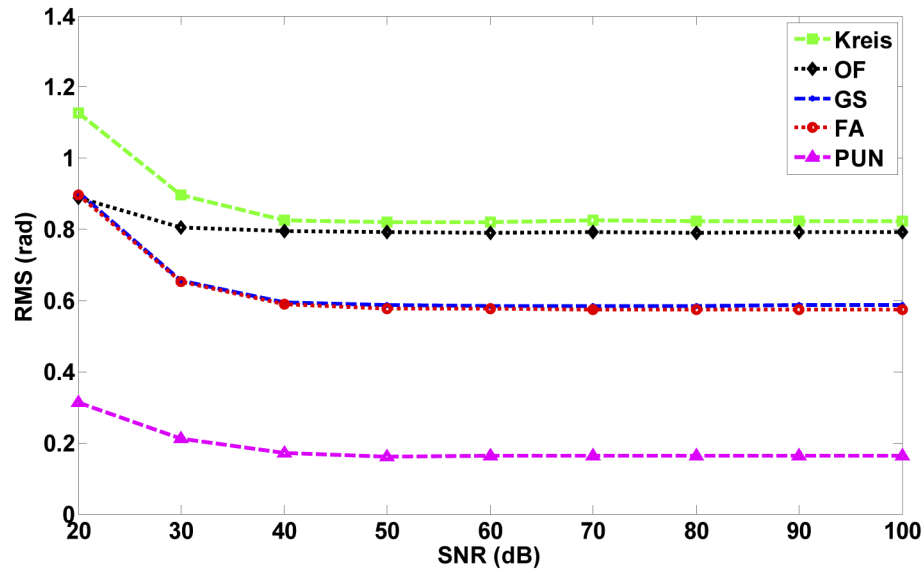


Fig. 8. RMS errors of the Kreis, OF, GS, FA and PUN when SNR changes from 20 dB to 100 dB with a phase step of 1 rad.

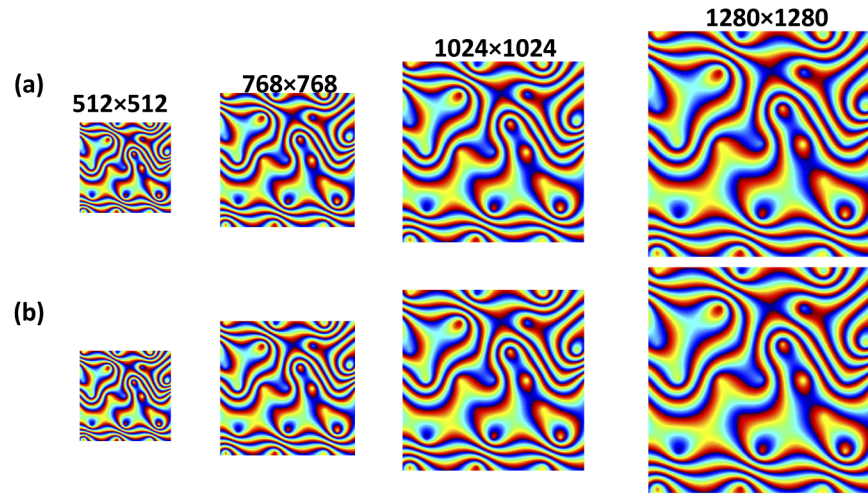


Fig. 9. An example of PUN results with different input sizes in Table 1. (a) the ground truth, (b) the PUN results.

Table 1. RMS errors and processing times of five methods with different image sizes with SNR=100 dB and phase step=1 rad

Methods	512 × 512		768 × 768		1024 × 1024		1280 × 1280	
	RMS(rad)	Time(s)	RMS(rad)	Time(s)	RMS(rad)	Time(s)	RMS(rad)	Time(s)
Kreis	0.8223	0.1153	0.8301	0.2968	0.8346	0.4695	0.8318	0.7136
OF	0.7915	0.6167	0.7676	1.3449	0.7579	2.3907	0.7573	3.8062
GS	0.5859	0.0293	0.5832	0.0664	0.5867	0.1164	0.5878	0.1931
FA	0.5751	0.0519	0.5712	0.0521	0.5749	0.0834	0.5743	0.1060
PUN	0.1625	0.3944	0.1536	0.7077	0.1302	1.2048	0.1302	1.8486

Table 2. RMS errors and processing times of five methods with different image sizes with SNR=20 dB and phase step=1 rad

Methods	512 × 512		768 × 768		1024 × 1024		1280 × 1280	
	RMS(rad)	Time(s)	RMS(rad)	Time(s)	RMS(rad)	Time(s)	RMS(rad)	Time(s)
Kreis	1.1211	0.1140	1.1822	0.2559	1.2108	0.4767	1.2309	0.7321
OF	0.8880	0.6178	0.8656	1.3487	0.8558	2.3959	0.8550	3.7199
GS	0.8988	0.0298	0.9024	0.0670	0.8991	0.1197	0.8971	0.1889
FA	0.8956	0.0509	0.8998	0.0596	0.8948	0.0894	0.8941	0.1472
PUN	0.3215	0.3769	0.3150	0.6876	0.2925	1.2074	0.3022	2.0128

calculated the wrapped phases using the standard four-step algorithm as the ground truth as shown in Fig. 10(b) and Fig. 12(b). The sizes of interferograms were 604×674 (we used a polarization camera PolarCamTM with the pixel number of 1208×1348 and the pixel size of 7.4 μm from 4D Technology, Inc. to take 4 pictures), so we extended them into 768×768 by padding zeros. The rest images in Fig. 10(b) and Fig. 12(b) are the wrapped phases calculated from the first two interferograms using Kreis, OF, GS, FA, and PUN. The RMS errors (comparing with the four-step algorithm) in Fig. 10(b) were 0.6485 rad, 1.2680 rad, 0.6590 rad, 0.7231 rad and 0.4007 rad respectively. And the RMS errors in Fig. 12(b) were 0.7080 rad, 1.0370 rad, 0.5610 rad, 0.5832 rad and 0.5019 rad respectively. Then we unwrapped the wrapped phases using the ‘unwrap’ function in the MATLAB. As shown in Fig. 11, comparing with the four-step interferometry, the RMS errors of Kreis, OF, GS, FA, and PUN after unwrapping were 0.7172 rad, 11.3122 rad, 0.2917 rad, 0.5061 rad and 0.1681 rad, respectively. And in Fig. 13, the RMS errors after unwrapping were 1.9243 rad, 6.5721 rad, 0.1975 rad, 0.8105 rad and 0.1688 rad respectively. Although the standard four-step interferometry cannot actually be the ground truth, it can be seen reliable regardless of the noise in interferograms. Therefore, the PUN results can be seen more accurate than the other two-frame interferometry algorithms.

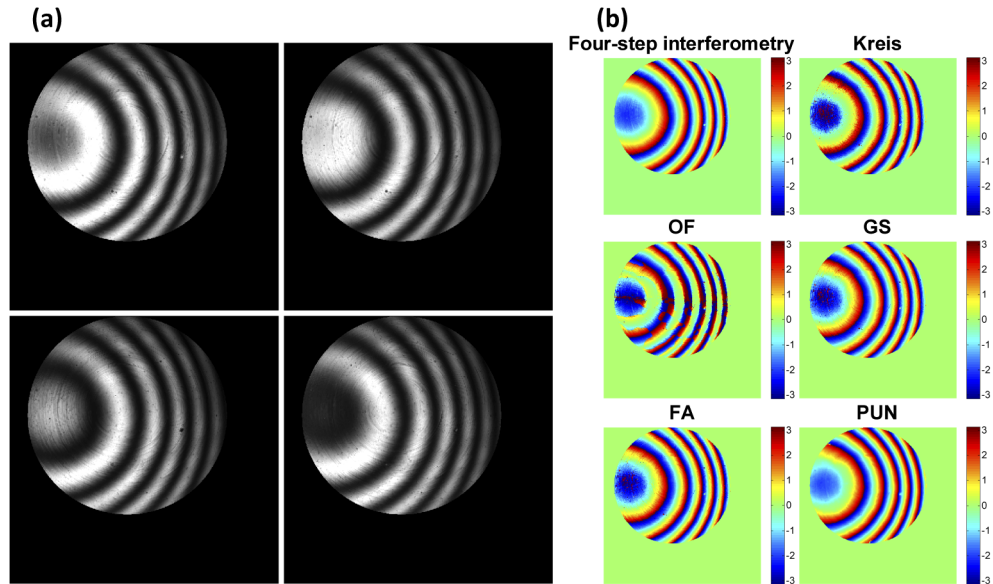


Fig. 10. Evaluation of different two-frame methods with the first set of experimental data. (a) Interferograms with $\pi/2$ phase shift, (b) the wrapped phase calculated by standard four-step interferometry, OF, GS, FA, and PUN.

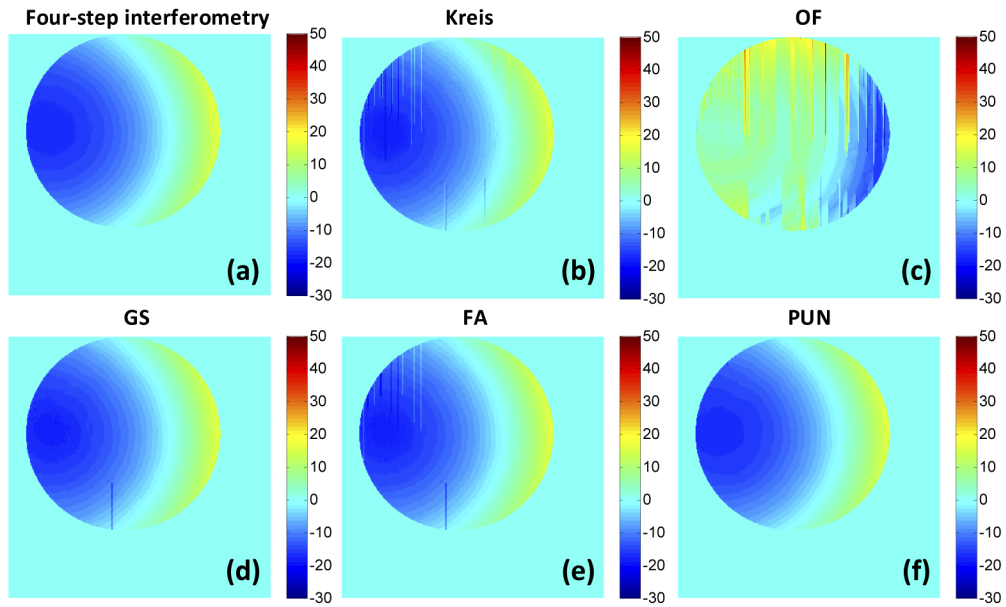


Fig. 11. Unwrapped phases of Fig. 9(b). (a) The standard four-step interferometry, and (b)-(f) Kreis, OF, GS, FA, and PUN.

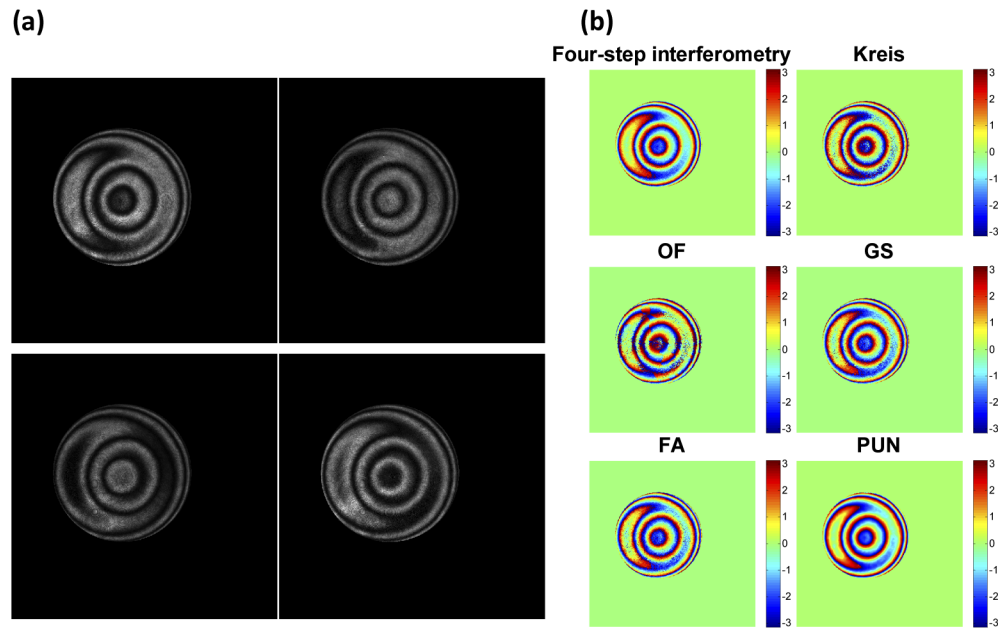


Fig. 12. Evaluation of different two-frame methods with the second set of experimental data. (a) Interferograms with $\pi/2$ phase shift, (b) the wrapped phase calculated by standard four-step interferometry, OF, GS, FA, and PUN.

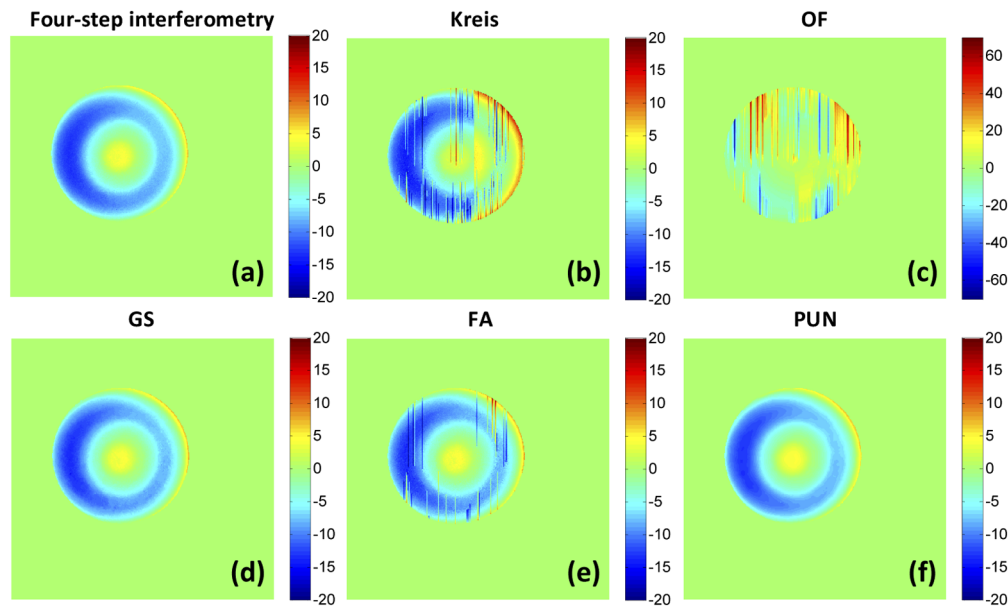


Fig. 13. Unwrapped phases of Fig. 11(b). (a) The standard four-step interferometry, and (b)-(f) Kreis, OF, GS, FA, PUN

5. Conclusion

In conclusion, we have proposed a new deep learning-based method called PUN to estimate the wrapped phase by using only two interferograms. The advantages of the PUN over one-frame methods and other two-frame methods have been explained in this paper. Admittedly, the proposed method is not the fastest comparing with other two-frame methods, however, it is indeed the most accurate. Moreover, deep learning-based methods normally need a fixed input size, our network architecture can use different input sizes such as 512×512 , 768×768 , 1024×1024 , 1280×1280 , or even larger after re-trained with new datasets. Both simulations and experiments have been done to verify the performance.

Funding

National Institutes of Health (S10OD018061); National Science Foundation (1918260); China Scholarship Council (201904910369).

Disclosures

The authors declare no conflicts of interest.

References

1. D. Malacara, *Optical Shop Testing* (Wiley, 2007).
2. G. Lai and T. Yatagai, "Generalized phase-shifting interferometry," *J. Opt. Soc. Am. A* **8**(5), 822–827 (1991).
3. Z. Wang and B. Han, "Advanced iterative algorithm for phase extraction of randomly phase-shifted interferograms," *Opt. Lett.* **29**(14), 1671–1673 (2004).
4. J. Deng, K. Wang, D. Wu, X. Lv, C. Li, J. Hao, J. Qin, and W. Chen, "Advanced principal component analysis method for phase reconstruction," *Opt. Express* **23**(9), 12222–12231 (2015).
5. M. Takeda, H. Ina, and S. Kobayashi, "Fourier-transform method of fringe-pattern analysis for computer-based topography and interferometry," *J. Opt. Soc. Am.* **72**(1), 156–160 (1982).
6. C. J. Evans and J. B. Bryan, "Compensation for Errors Introduced by Nonzero Fringe Densities in Phase-Measuring Interferometers," *CIRP Ann.* **42**(1), 577–580 (1993).

7. H. Yiwei, X. Hou, Q. Haiyang, and W. Song, "Retrace error reconstruction based on point characteristic function," *Opt. Express* **23**(22), 28216–28223 (2015).
8. T. M. Kreis and W. P. O. Juptner, "Fourier transform evaluation of interference patterns: demodulation and sign ambiguity," *Proc. SPIE* **1553**, 263–273 (1992).
9. J. Vargas, J. A. Quiroga, C. O. S. Sorzano, J. C. Estrada, and J. M. Carazo, "Two-step interferometry by a regularized optical flow algorithm," *Opt. Lett.* **36**(17), 3485–3487 (2011).
10. J. Vargas, J. A. Quiroga, C. O. S. Sorzano, J. C. Estrada, and J. M. Carazo, "Two-step demodulation based on the Gram-Schmidt orthonormalization method," *Opt. Lett.* **37**(3), 443–445 (2012).
11. Z. Cheng and D. Liu, "Fast and accurate wavefront reconstruction in two-frame phase-shifting interferometry with unknown phase step," *Opt. Lett.* **43**(13), 3033–3036 (2018).
12. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, (Springer, 2015), pp. 234–241.
13. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition* (IEEE, 2017), pp. 1125–1134.
14. K. Wang, Y. Li, Q. Kema, J. Di, and J. Zhao, "One-step robust deep learning phase unwrapping," *Opt. Express* **27**(10), 15100–15115 (2019).
15. J. C. Zhang, X. B. Tian, J. B. Shao, H. B. Luo, and R. G. Liang, "Phase unwrapping in optical metrology via denoised and convolutional segmentation networks," *Opt. Express* **27**(10), 14903–14912 (2019).
16. K. Yan, Y. Yu, T. Sun, A. Asundi, and Q. Kema, "Wrapped phase denoising using convolutional neural networks," *Opt. Laser Eng.* **128**, 105999 (2020).
17. D. Kando, S. Tomioka, N. Miyamoto, and R. Ueda, "Phase Extraction from Single Interferogram Including Closed-Fringe Using Deep Learning," *Appl. Sci.* **9**(17), 3529 (2019).
18. D. Malacara, M. Servin, and Z. Malacara, *Interferogram analysis for optical testing* (CRC press, 2005).
19. Z. Ge, F. Kobayashi, S. Matsuda, and M. Takeda, "Coordinate-transform technique for closed-fringe analysis by the Fourier-transform method," *Appl. Opt.* **40**(10), 1649–1657 (2001).
20. L. Huang, C. Rao, and W. Jiang, "Modified Gaussian influence function of deformable mirror actuators," *Opt. Express* **16**(1), 108–114 (2008).
21. A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the International Machine Learning Society*, 30 (2013).
22. X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (2011), pp. 315–323.
23. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv:1502.03167 [cs] (2015).
24. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
25. T. U. Djork-Arné Clevert and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," arXiv:1511.07289 [cs.LG] (2015).
26. D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, (San Diego, 2015).