Unsupervised feature learning and clustering of particles imaged in raw holograms using an autoencoder

ZONGHUA LIU[1,*], THANGAVEL THEVAR[2], TOMOKO TAKAHASHI[3], NICHOLAS BURNS[2], TAKAKI YAMADA[4], MEHUL SANGEKAR[3], DHUGAL LINDSAY[3], JOHN WATSON[2], AND BLAIR THORNTON[1,4]

[1] *Institute of Industrial Science, University of Tokyo, Tokyo 153-8505, Japan*
[2] *School of Engineering, University of Aberdeen, Aberdeen AB24 3FX, U.K.*
[3] *X-STAR, JAMSTEC, Yokosuka 237-0061, Japan*
[4] *Centre for In Situ and Remote Intelligence, Faculty of Engineering and Physical Sciences, University of Southampton, Southampton SO17 1BJ, U.K.*
[*] *Corresponding author: zonghua@iis.u-tokyo.ac.jp*

Compiled August 31, 2021

Digital holography is a useful tool to image microscopic particles. Reconstructed holograms give high-resolution shape information that can be used to identify the types of particles. However, the process of reconstructing holograms is computationally intensive and cannot easily keep up with the rate of data acquisition on low-power sensor platforms. In this work, we explored the possibility of performing object clustering on holograms that have not be reconstructed, *i.e.* images of raw interference patterns, using the latent representations of a deep-learning autoencoder and self-organising mapping network in a fully unsupervised manner. This concept was demonstrated on the synthetic raw holograms achieving the clustering accuracy of 94.4%. This was close to 97.4% of the accuracy achieved using their reconstructed holograms, reducing the computational time by three orders of magnitude. It takes around 0.09 second to process a hologram on a low-power CPU board using the proposed method, which makes it possible to carry out clustering interpretation in real time on low-power sensor platforms. Experiments were also performed on real holograms. For the real raw holograms for testing, the clustering accuracy was 47.1% when the models were trained only on the real raw training data. The accuracy increased to 64.1% when the models were entirely trained on the synthetic raw training data. The highest accuracy of 75.9% was achieved when the models were trained on the both datasets using transfer learning. Regarding the reconstructed holograms, the lowest accuracy was 58.4% obtained when the models only trained on the real data. It increased to 70.2% when the model only trained on the synthetic data. However, transfer learning did not result in an increase of accuracy in the reconstructed holograms in our work.

http://dx.doi.org/10.1364/ao.XX.XXXXXX

1. INTRODUCTION

Holography is a non-invasive high-resolution imaging technique that retains a large depth-of-field [1]. Digital holographic microscopes can be used to generate focused images of microscopic particles that are suspended in fluids, such as marine microparticles [2–4] and biological cells *in vivo* [5, 6]. Since a raw hologram consists of the interference pattern generated when a particle is in the path of a coherent light, it is normally necessary to first reconstruct the hologram at a specific distance (the focused reconstruction) so that the particle's shape can be clearly seen before any further analysis, like object classification, can be performed. However, hologram reconstruction is a computationally intensive process. It becomes more expensive when the specific distance is unknown prior to reconstruction, since hologram reconstruction needs to go through the whole recording volume to detect the focal plane. Although efforts have been made to speed up this process using field-programmable gate arrays (FPGAs) [7, 8] and parallel processing using graphics cards [9], these methods significantly increase the cost, power consumption and complexity of embedded sensing platforms.

Recent demonstrations of supervised deep-learning techniques to efficiently reconstruct raw holograms [10–12] give the possibility for real-time interpretation of digital holograms on compact, low-power devices. However, the need for large training datasets is a limiting factor because reconstruction and focus detection in holograms is time consuming. At the same time, the fact that deep-learning algorithms can extract useful features from raw holograms motivated our investigation into direct interpretation using deep-learning autoencoders [13, 14]. A key feature of autoencoders is that they can learn latent representations in a fully unsupervised manner (*i.e.* without the need for any human input to generate training data), which greatly simplifies the training process. Unlike traditional methods for representation extraction, *e.g.* principal component analysis (PCA) [15], autoencoders are capable of modelling more sophisticated and complex nonlinear relationships between inputs and their representations [16]. Therefore, autoencoders can be easily redeployed and retrained on data gathered under different conditions or using a different instrument. Latent representations extracted by autoencoders can be used for object clustering without the need for any human supervision, and this method has been effectively demonstrated using other types of optical image [17–19]. However, there have been no previous studies investigating their use for clustering of raw digital holograms.

In this paper, we explore the possibility of using an end-to-end unsupervised workflow to extract the features from raw holograms and then cluster them based on these features. Even though unsupervised methods do not require human input to generate labelled training data, they still require large amounts of unlabelled data to learn useful latent representations, which can be challenging to obtain in applications (*e.g.* marine microparticle imaging). Therefore, we investigate how to improve the efficiency of training unsupervised models using synthetic data. The concept of directly interpreting raw holograms is first demonstrated entirely using synthetic holographic data. Next, we explore transfer learning [20], where models are pre-trained on synthetic holograms, and the pre-trained models are trained on a small number of real holograms. We also demonstrate the proposed workflow on a low power CPU board to show its practical usefulness for in situ applications.

2. AUTOENCODERS

An autoencoder consists of two components: an encoder and a decoder, as shown in Fig. 1. The encoder reduces an input image x into a latent representation h that has a lower number of dimensions than the original image. The decoder does the reverse, using the latent representation h to restore[1] the input image to x_r that is as close to the initial input as possible. It is often useful to add noise to the inputs so that the encoder learns to denoise images, which aids to extract robust representations from inputs [14, 21].

The model learns through minimising the difference, or loss, between the inputs and outputs for a set of images, *i.e.* the training data. The process can be described as follows:

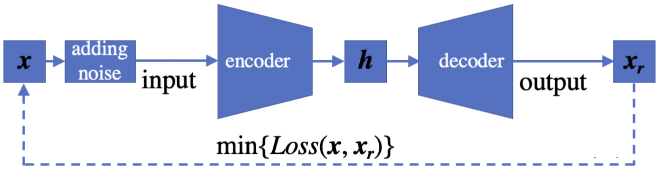

Fig. 1. Flowchart of an autoencoder with denoising. x, h and x_r signify an input image, latent representation and reproduced image respectively. $Loss(x, x_r)$ indicates the loss function which calculates the error between x and x_r.

$$\{\varphi : x \to h; \phi : h \to x_r; \varphi, \phi \Leftarrow min(Loss(x, x_r))\} \quad \textbf{(1)}$$

where φ and ϕ signify the transition of the encoder and decoder respectively. The training attempts to find the optimal weights in φ and ϕ to minimise the loss between x and x_r. Once trained, the encoder can be used independently to extract latent representations that can be used as features for clustering or classification.

3. LEARNING MODELS AND DATASETS

A. Autoencoder

The autoencoder architecture used in this work is based on the AlexNet neural network [22]. This model is effective in describing images, and won the ImageNet Large Scale Visual Recognition Challenge in 2012 [23]. The original architecture of AlexNet consists of 8 layers in total, taking input image dimensions of 227 × 227 × 3, using 5 convolutional layers (the first, second and fifth layer is followed by a max pooling layer respectively) and 3 fully-connected layers. The relatively simple architecture compared to more recent CNN makes it suitable for use in autoencoders, as demonstrated in [19, 24].

In this work, two modifications have been made to the original AlexNet architecture. The architecture of the modified autoencoder is described in Section 1 of the supplemental document. Since typical holographic images are monochrome, the input data size is changed to 227 × 227 × 1 instead of 227 × 227 × 3, which caters for the RGB colour channels in conventional imaging. The three fully-connected layers in the original take up 94% of the parameters and are useful for solving highly complex classification problems [25]. The fully-connected layers ignore the image structure and their output features lose geometric characteristics of the input images [26], while the convolutional layers share their weights amongst all locations in the input and preserve spatial locality [22]. Since raw holograms have a high degree of geometric structure (interference fringes around object silhouettes), we replaced the three fully connected layers by two convolutional layers (followed by a max pooling layer respectively). This convolutional modification is able to not only facilitate feature extraction and improve the results in our work, but also speed up the training process and reduce the network's size (the details have been shown in Section 3-B of the supplemental document).

In the first modified convolutional layer, the number of filters used is 96, with a kernel size of 3 × 3, and scanning strides of 1 × 1. The "same padding" strategy is used in this layer. Therefore, this layer outputs a datum in the size of 6 × 6 × 96. After max pooling with the pooling size of 3 × 3 and the scanning stride of 3 × 3, the output datum size becomes 2 × 2 × 96. The second convolutional layer is designed to control the number

[1] To clarify the term of reconstruction (reconstructed image), in this paper, the output of the autoencoder is called restoration (restored image from the input); the output of the hologram reconstruction algorithm is called reconstruction (reconstructed image from the input).

of the latent features. Its output size is 2 × 2 × 40. A ReLU (rectified linear unit) activation function is used in these two convolutional layers. After max pooling, a 40-dimension latent representation of an input image is obtained. This value was chosen based on a parametric study, where increasing the dimensionality of latent representation did not improve the results (see Section 3-A in the supplemental document). Its decoder is mirror-symmetrical, where convolutional layers are transposed to transconvolutional layers [27], and the max pooling layers are transposed to upsampling layers [28].

Since much background noise exits in holograms [29], the functionality of denoising is added to the autoencoder to reduce the effect of noise on feature extraction (Section 3-C in the supplemental document). The training parameters for the autoencoder are described in Section 2-A of the supplemental document.

B. Clustering model

In this work, objects are clustered using a self-organising mapping (SOM) network [30]. The SOM is a well-known classical unsupervised learning model, and it is simple to implement [31]. This model is built using a pre-defined 2-D net of neurons. Unlike the error-correction-based learning in other networks (*e.g.* gradient descent in backpropagation), competitive learning [30] is applied where training samples compete for neurons to represent them. This causes different portions of the SOM network to respond similarly to certain input samples, creating a transfer function where similar regions of the latent representation will be mapped to the same cluster. Further details of the SOM used can be found in Section 2-B of the supplemental document.

C. Datasets

In applications such as marine micro-particle imaging, it can be difficult to prepare massive real holographic data for training a deep-learning autoencoder. One possible solution is to create a set of synthetic holograms and use these to pre-train a model. Afterwards, the pre-trained model can be used as the starting point for further training on a small quantity of real data using the technique of transfer learning. Since it is easy to add/remove artificial noise into/from synthetic holograms, pre-training the autoencoder on synthetic holograms also facilitates the training process with denoising.

Experiments were performed on both raw interference patterns, and reconstructed images of four simple geometries: circle, triangle, rectangle, diamond. A 200 mm × 200 mm glass plate with these shape patterns etched on it with about 1 mm separation between them was used as a target to record real holograms. The diameter of the circle and the smallest edge of other patterns is 100 µm, as shown in Fig. 2. When creating the synthetic dataset, the shapes do not have any neighbours.

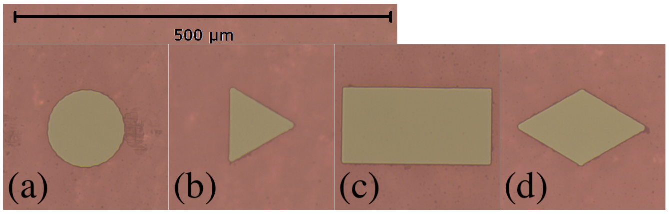

Fig. 2. Microscopic photographs of four shapes. (a) – circle; (b) – triangle; (c) – rectangle; (d) – diamond.

Real dataset: An in-line holographic camera, shown in Fig. 3, was used to take holograms of the shape plate. A 532 nm, single-longitudinal mode continuous wave laser (Elforlight) was used as the light source. The beam intensity was controlled using a variable neutral density filter, while a spatial filter (items ③ and ④ in Fig. 3) provided a spatially coherent and uniform beam. This beam was then collimated by a lens before illuminating the CMOS image sensor (JAI GO-5100-USB) which has a resolution of 2464 × 2056 with a pixel pitch of 3.45 µm × 3.45 µm, giving an active area of 8.5 mm × 7.09 mm.

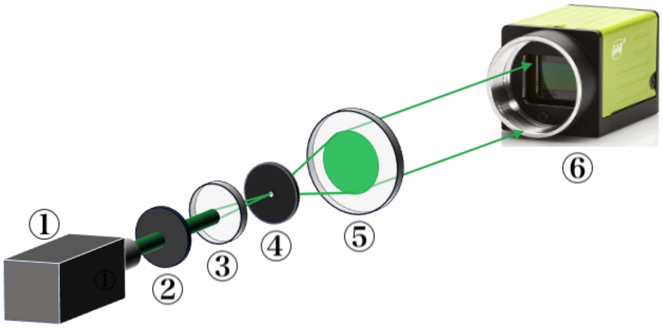

Fig. 3. Schematic diagram of the in-line structure hologram recorder used in this work. ① – laser, ② – neutral density filter, ③ – microscopic objective lens, ④ – pinhole, ⑤ – collimating convex lens, ⑥ – CMOS image sensor.

The shape plate was placed in the laser beam path, between the collimating lens and the sensor. Its distance from the sensor was varied between 10 mm to 60 mm along with different sensor exposure times (10, 40, 70, 100, 130, 160, 190 and 220 µs) and plate orientation to the plane of the sensor. Fig. 4 shows four holograms of the rectangle recorded under different conditions.

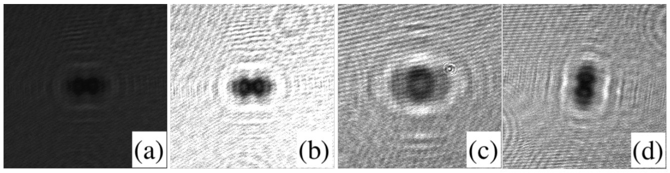

Fig. 4. Four hologram samples of a rectangle under different conditions. (a) recorded at 17.90 mm with 10 µs exposure time; (b) recorded at 17.90 mm with 220 µs exposure time; (c) recorded at 47.70 mm with 130 µs exposure time; (d) recorded at 17.85 mm with 130 µs exposure time and close to 90° rotation with regard to positions in the other three holograms.

Two groups of real holographic data were collected. One of them (Group 1) was used to further train the pre-trained autoencoder, and the other (Group 2) was used to test the model. Each hologram was cropped to 300 × 300 pixels around the target (the reason is given in Section 3-D of the supplemental document), resulting in 4,180 cropped holograms in Group 1 and 3,844 in Group 2 (see Table 1). They were reconstructed using the angular spectrum method [32], with examples of reconstructed holograms shown in Fig. 5.

Synthetic dataset: A shape image was first created, and its hologram was simulated using the angular spectrum method [32]. The parameters used for the simulation are shown in Table 2. The size and recording distance of the shape are randomly selected from the given ranges. Shape's centre and orientation are also randomly chosen, but are restricted so that the shape is fully shown within the boundary of the image.

Table 1. Number of real holograms for each shape.

Shape		circle	triangle	rectangle	diamond	in total
Number	Group 1	780	887	1522	991	4,180
	Group 2	891	708	1546	699	3,844

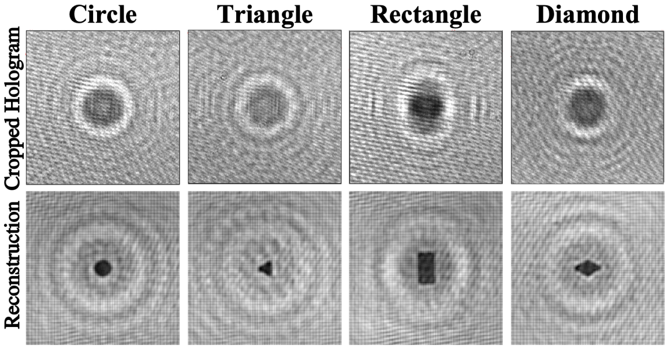

Fig. 5. Cropped holograms of four shapes with the size of 300 × 300 and their reconstructions.

Table 2. Parameters used to create the synthetic holographic dataset.

Parameters	Values
shape size (μm)	50 – 300 with interval of 1
image size (pixel number)	227 × 227
wavelength (nm)	532
pixel pitch (μm)	3.45 × 3.45
recording distances (mm)	10 – 60 with interval of 0.5

In this dataset, three groups of data were created: training data consisting of 24,000 holograms, validation data with 8,000 holograms and testing data with 16,000 holograms. In each group, the number of each shape was equal. The histogram of the recording distances (in five ranges) and shapes' sizes (in five ranges) in the three groups is shown in Fig. 6. Regarding the recording distance, the number of the holograms of each shape in each range is similar. Based on the size, most of holograms lie within the range of 100 – 250 μm, which accords with the shapes' size situation in the glass plate (see Fig. 2).

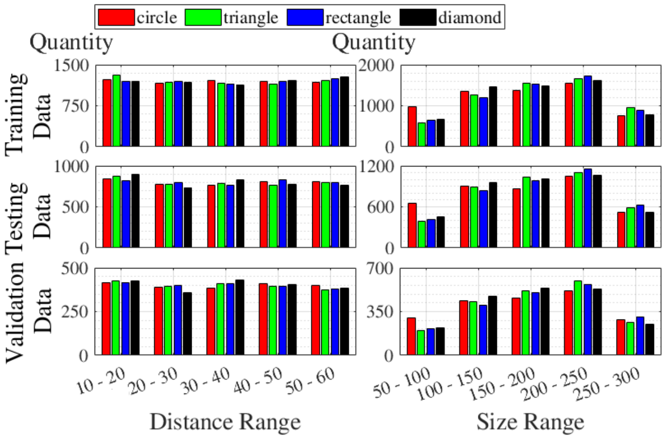

Fig. 6. Histogram of recording distances and shapes' sizes in three groups.

The data of both raw and reconstructed holograms are generated using the angular spectrum method. Two examples in each shape are shown in Fig. 7, with the original shapes, the synthetic holograms and their reconstructions.

To simulate more realistic holograms, noise was added by taking real holograms without any targets and superimposing randomly cropped regions of them as background noise in the synthetic holograms (see Fig. 8). This process can also facilitate training the autoencoder with the functionality of denoising.

4. EXPERIMENTAL RESULTS AND ANALYSIS

The clustering performance of the proposed method is verified on the raw holograms of the entirely synthetic, entirely real and combined hologram datasets, and the results are compared to the equivalent performance for the reconstructed holograms. In the first set of experiments, both training and evaluation were only performed on the synthetic data. Next, the experiments were performed on the real dataset. The pre-trained autoencoders on the synthetic holograms (raw and reconstructed holograms respectively) were further trained using the corresponding real data in Group 1. Afterwards, their encoders were used to extract the latent features from the corresponding real data in Group 2, and these features were used to cluster these real holograms. For comparison, we also performed training using only real data.

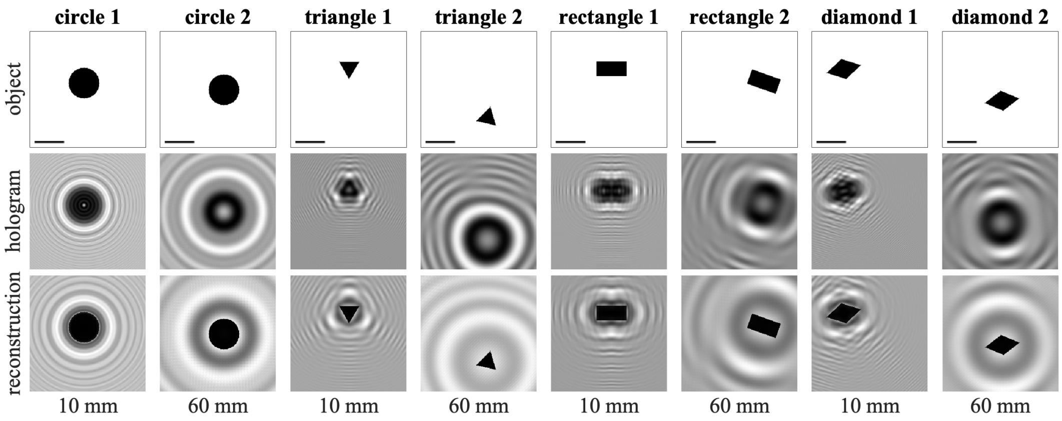

Fig. 7. Two examples of each shape, including original shapes (in the first row), corresponding synthetic holograms (in the second row) and their reconstructions (in the third row). Number below each column gives the recording distance of the hologram. The scale lines in the first row indicate 200 μm.

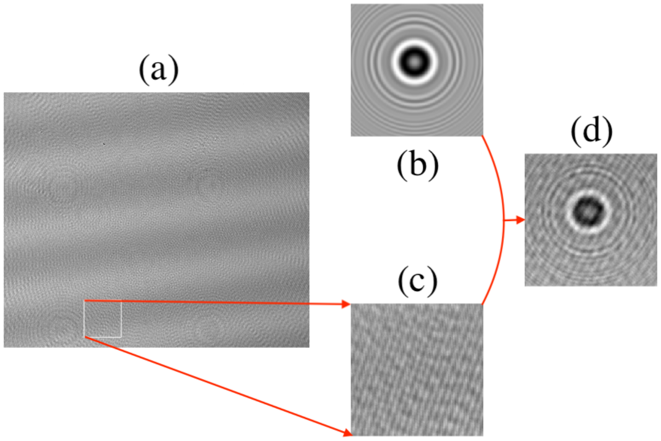

Fig. 8. An example of adding noise to the synthetic hologram. The noise image (c) is cropped from a background hologram (a), and it is added to a synthetic hologram (b) to create the final synthetic hologram (d).

The clustering performance was assessed using the overall accuracy and F1 score [33, 34] compared to the ground truth and the computational runtime. The workstation used for training the models had an Intel i9-9900K CPU @ 3.60 GHz × 16 with 36 GB RAM and a GPU of NVIDIA GeForce RTX 2080 with 8 GB RAM. The low-power CPU board used to run the proposed models had an Intel Atom processor E3940 @ 3.60 GHz × 4 with 8 GB RAM, which could be directly integrated into a compact digital holographic microscope.

Python was used to interpret all the algorithms discussed in this work. The angular spectrum algorithm [32] was used to reconstruct a hologram at a given distance, and the autofocusing method described in [35] was used to detect the focused reconstruction across the entire recording distance range. Unless an output focused reconstruction looked obviously wrong, human was not involved to refine the result. In order to speed up the algorithms of angular spectrum and autofocusing, two Python-based modules were used: **mpi4py-fft** [36] for parallelly computing fast Fourier transforms in the algorithm, and **multiprocessing** [37] for parallelising the execution of reconstruction across the recording distance range. The autoencoder was developed, trained and tested using **Tensorflow** [38]. The SOM model was built, trained and tested using the open-source library of **MiniSom** [39].

A. Feature extraction and object clustering on synthetic holograms

The clustering performance of the proposed method was first evaluated using the synthetic holograms. The autoencoder and SOM were trained on the synthetic training data (raw and reconstructed holograms respectively). Afterwards, each pair of the trained encoder and SOM were used to cluster the corresponding raw and reconstructed datasets for testing.

Fig. 9 shows the loss of the autoencoder on the training dataset (24,000 holograms) and validation dataset (8,000 holograms) in 100 epochs. The fact that the loss is similar for training and validation indicates that the model was able to generalise, and it was not over-fitting the synthetic data. The result also shows that convergence was achieved after ~40 epochs.

Fig. 10 shows the TSNE [40] plots of the latent representations extracted from the raw and reconstructed holograms for the testing data by the corresponding trained encoders. It shows that there are clearer separations between the points indicating different shapes in the reconstructed holograms, while some merging between different shapes occurs in the plot of the raw data. This could be reflected in the clustering scores of these shapes that would be lower in the raw holograms than the reconstructed holograms. Besides that, some points of shape circle appear in shape triangle in the raw data, and this would result in lower scores in these two shapes.

The autoencoder and SOM were trained five times, and each pair of trained encoder and SOM were used to cluster the corresponding raw and reconstructed datasets for testing. The clustering performance of the SOM was compared to two different classification methods. It should be noted that while the SOM can cluster the dataset in a fully unsupervised manner, the classifiers used for comparison both required human expert labelled training data (in this case this is the known ground truth of the synthetic data) to determine the shape of the targets. The first was a support vector machine (SVM) [41] that was trained on the features extracted from the training data by the encoder. This was then used to classify the test data (training parameters are given in Section 2-C of the supplemental document). The

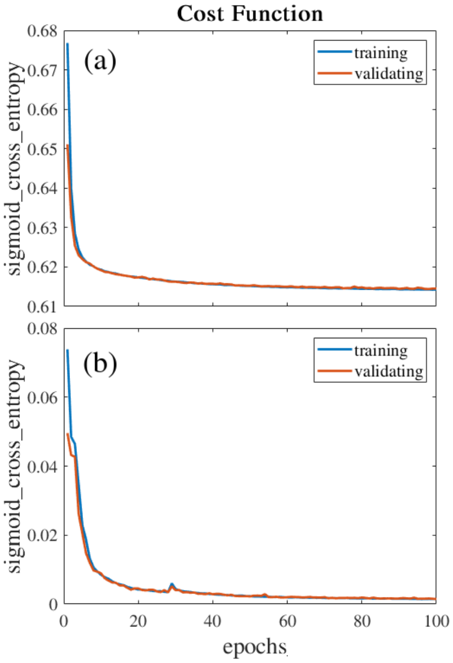

Fig. 9. Cost curves of the autoencoder in the processes of training and validation on the raw (a) and reconstructed (b) synthetic holograms. Each cost value is the mean of the results from five experiments.

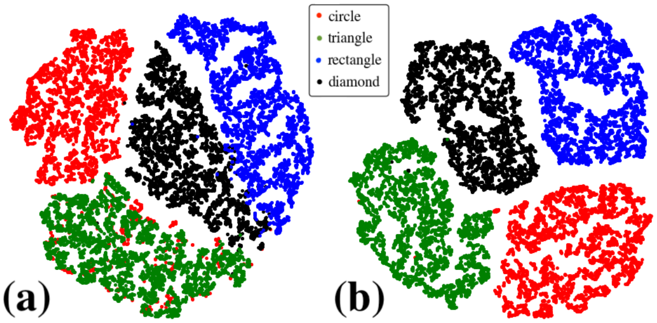

Fig. 10. Distribution of 2D feature data (compressed from the representations extracted by the encoder) in the raw (a) and reconstructed (b) synthetic testing holograms using TSNE.

second method used AlexNet[1] to directly classify the input images based on the labelled training data. Table 3 shows their performance for the raw and reconstructed holograms in the testing data. The clustering accuracy of the proposed method reached 94.4% and 97.3% respectively. The corresponding F1 score of each target was also lower for the raw holograms than the reconstructed holograms. The two supervised classifiers used achieved higher F1 accuracy scores than the proposed unsupervised clustering using the SOM. This is to be expected, since labelled training data is provided to the classifiers. The main advantage of the unsupervised approach is that it does not require any human labels for training, which is generally time-consuming to generate and is challenging for applications where the exact target classes in the dataset are not initially known. An interesting observation is that the SVM classifier achieved close to 100% accuracy using the same features as the SOM. This indicates that it is the SOM that limits clustering performance and not encoder.

Table 4 shows the time taken for the different computations carried out in the experiment. The autoencoder and SOM were trained on the workstation, and testing the trained models was done on the low-power CPU board. The time for training the autoencoder and SOM was almost identical for the raw and reconstructed holograms. The biggest cost was in the reconstruction of the holograms, which took more than 13 times the combined training time. This highlights the advantage of using raw holograms, which does not require this step. Clustering the entire testing dataset consisting of 16,000 images using each pair of trained encoder and SOM took around 1,500 s, or ~0.09 s to process one hologram on average. This processing speed is high enough to carry out real-time clustering on the lower-power CPU board for an image acquisition rate of less than 10 Hz. However, reconstructing each hologram on the lower-power CPU board took ~14 s, which makes real-time clustering of reconstructed holograms impossible. It should be noted that hardware optimisation, such as the use of FPGAs or GPUs embedded single board computers as demonstrated by [7–9], can allow real-time reconstruction at faster rates. However, this comes at the cost of higher power consumption, which is not ideal for many low-power, long term monitoring applications.

B. Feature extraction and object clustering on real holograms

In this experiment, the autoencoder and SOM were first trained on the synthetic training holograms, and the pre-trained autoencoder was trained on a small group of real holograms (Group 1) using transfer learning (see Section 2-D in the supplemental document). The pre-trained SOM was also trained using the features of the holograms in Group 1 extracted by the re-trained autoencoder [2]. Afterwards, the final trained encoder and SOM was used to extract and cluster the latent representations from the other group of real holograms for testing in Group 2.

Latent representation extraction: The real holograms for testing were fed to the final trained autoencoder and SOM as mentioned above. For comparison, three other sets of experiments were carried out: C1. the autoencoder was trained on the ImageNet dataset[3] (2012 [23]) and the real holographic training data (transfer learning); the SOM was trained on the real training data based on their features extracted by the trained encoder; C2.

[1] The image input size is changed to 227 × 227 × 1 instead of 227 × 227 × 3. Its output class number is changed to 4. The training parameters are the same with those used to train the autoencoder.

[2] The parameters for re-training keep the same with those used in pre-training.

[3] The images were converted into grayscale.

Table 3. Results of the three methods based on F1 score and accuracy when used to cluster/classify the synthetic testing holograms.

	Shape	encoder+SOM		encoder+SVM		AlexNet	
		F1 Score	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy
Raw Holograms	circle	0.933	94.4%	0.980	98.9%	1.000	99.8%
	triangle	0.930		0.980		1.000	
	rectangle	0.966		1.000		1.000	
	diamond	0.948		1.000		1.000	
Reconstructed Holograms	circle	0.975	97.4%	1.000	99.9%	1.000	100.0%
	triangle	0.978		1.000		1.000	
	rectangle	0.980		1.000		1.000	
	diamond	0.962		1.000		1.000	

Note: Each value is the mean of the results from five experiments.

Table 4. Performance of running time when the models used to extract features from raw and reconstructed holograms and cluster them.

	Time (s) [a]				
	reconstruction for training [b]	autoencoder training	SOM training	reconstruction for testing [b]	clustering for testing
Raw Holograms	-	3,229	3.8	-	1472
Reconstructed Holograms	42,240	3,235	3.9	226,240	1477

[a] average value of five experiments.
[b] image size: 227 × 227; reconstruction distance range: 10 – 60 mm with step 0.1 mm; no manual operation included.
Note: Training was carried out on the workstation and testing was done on the CPU board.

the autoencoder and SOM were trained only on the synthetic training data; C3. the autoencoder and SOM were trained only on the real training data. The description on these four sets of experiments are given in Table 5.

The latent representations of the real testing holograms extracted by the encoders from these four experiments can be visualised using the TSNE, as shown in Fig. 11. Compared with the TSNE plots of the synthetic data shown in Fig. 10, their distributions show significantly decreased separations between the points indicating the different shapes and a separation between the points indicating the same shape (rectangle). Normally, a bad distribution of representations in the TSNE tends to correspond with a low clustering result in the representations. The encoder trained only on real training data (Fig. 11-(d)) performed the worst both on the raw and reconstructed data, and points indicating different classes mixed together except for class rectangle. The results from the encoder trained only on synthetic training data (in experiment C2) became better, as shown in Fig. 11-(c). In experiment C1, the plots of the representations from the encoder trained on the ImageNet data and real holograms for training, shown in Fig. 11-(b), looked better than experiment C3, but worse than experiment C2. Regarding raw holograms, the encoder trained on the synthetic and real data in experiment P performed the best, as expected. Beyond expectation, however, the plot of the reconstructed holograms, was not as good as the corresponding plot in experiment C2, except for in class rectangle. One possible reason can be found through observing Fig. 12, which shows two output images of each shape restored by the autoencoders trained on the synthetic and real holographic data, and only synthetic data respectively. The autoencoder trained only on the synthetic reconstructed data with denoising allows it to restore reconstructed holograms with clear shape outlines, but re-training the model on the real reconstructed holograms reduces this capability. While this did not happen to the raw holograms. Conversely, the restored images from the autoencoder trained on the synthetic and real raw holograms show more similar details with their original inputs than the corresponding images from the autoencoder only trained on the synthetic holograms, such as the restored images of the two circles (the patterns in the images in the second row look like interference fringes of circles, while the patterns in the third row look like fringes of triangles rather than circles).

Clustering: Clustering of the real testing holograms in Group 2 was carried out where the encoder and corresponding SOM from those four sets of experiments described in Table 5 were used respectively. The accuracy and F1 score of each class were summarised in Table 6. When the models were trained only on the real data (experiment C3), the raw holograms achieved the accuracy of 47.1% and the reconstructed holograms achieved 58.4%. When the models were trained only on the synthetic data (experiment C2), the former accuracy became 64.1% and the latter became 70.2%. In the other two sets of experiments where transfer learning was used, the accuracy achieved in the raw holograms obviously increased to a value of ~76%, while the accuracy in the reconstructed holograms (~68%) did not change as much as in the raw holograms, especially compared with the value of 70.2% obtained in experiment C2. Regarding accuracy, the models trained on the synthetic and real data in experiment P had similar performance with the models trained on the ImageNet and real data in experiment C1. This was unexpected, as the TSNE plots of the latter were not as good as the former's (see Fig. 11). It implies that the SOM used was flexibly compensating and resulted in a good clustering accuracy. Since there are only 24,000 images in the synthetic training data, while there are 1,281,167 images in the training dataset of ImageNet 2012, there is still a benefit to pre-train the autoencoder using the synthetic data, although the similar results were obtained in those two sets of experiments. Another unexpected result is that the accuracy in the raw holograms was higher than the reconstructed holograms after using transfer learning. This has been reflected in Fig. 12, which shows that transfer learning did not facilitate the encoder to extract better representations from reconstructed holograms.

It should be noted however, that the performance across classes was not uniform based on F1 score in each set of experiments. The rectangles were always resolved the best, and the circles were resolved the worst both in the raw and reconstructed holograms. After using transfer learning, the circles and diamonds were better resolved in the raw holograms than the reconstructed holograms. The corresponding confusion matrices of the raw and reconstructed holograms from experiment P are shown in Fig. 13. In the raw holograms, it can be observed that there is obvious mis-identification between the classes of circle and triangle which causes low F1 scores in these two classes. One reason could be found in Fig. 11, where the restored patterns of the circles look similar with the triangles', which could result in the lowest F1 score in the class of circle. In the reconstructed holograms, a bigger mis-identification ratio occurs between the classes of circle and diamond and this causes lower F1 scores in them.

5. CONCLUSIONS

Object clustering can be efficiently performed on raw holograms to achieve comparable performance to equivalent reconstructed holograms. This offers significant gains in computational efficiency, which is compelling for *in-situ* applications where real-time interpretation cannot keep up with the rate of data acquisition. The key findings are:

- Deep-learning autoencoders can be used to extract latent representations from both raw and reconstructed holograms in a fully unsupervised manner. When using an SOM as a clustering model, the accuracy of the raw and reconstructed holograms achieved 94.4% and 97.4% respectively for the synthetic dataset generated in this work. While the accuracy is nearly 100% both in the raw and reconstructed holograms when an SVM is used as a classifier to classify the same dataset. This reflects that the proposed autoencoder has the capability to extract good representations from raw holograms, and the clustering performance limited by the SOM that was used for unsupervised clustering.
- A three-order gain in computational efficiency can be achieved by directly interpreting raw holograms compared to reconstructed holograms using the same processing hardware. It takes ~0.09 second on average to process a hologram on a low-power CPU board. This makes it possible to interpret holograms in real time when data are collected by a low-power sensor platform.
- Synthetic data can be used to train autoencoder-based clustering of real holograms. Comparing with the results from the synthetic data, the accuracy reduces to 64.1% and 70.2% for the real raw and reconstructed holograms respectively, which is better than the results from the models trained only on the real training holograms. Gains in performance happen through the use of the established transfer learning technique. After training the models on the synthetic and real training data, the accuracy increases to 75.9% in the raw holograms, but the accuracy hardly

Table 5. Description of four sets of experiments.

	Experiment	data for training autoencoder	data for training SOM	testing data
proposed method	P	synthetic^a+real (Group 1 ^{b})	synthetic+real (Group 1)	real (Group 2 ^{b})
comparative method	C1	ImageNet+real (Group 1)	real (Group 1)	real (Group 2)
	C2	synthetic	synthetic	real (Group 2)
	C3	real (Group 1)	real (Group 1)	real (Group 2)

[a] synthetic data for training.

[b] Group 1: real data for training; Group 2: real data for testing. See Table 1.

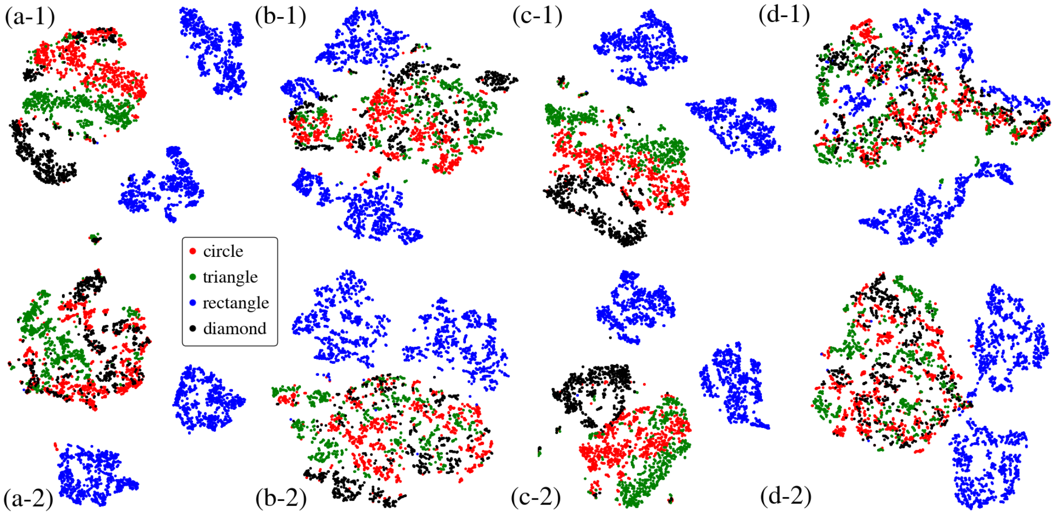

Fig. 11. Distribution of 2D feature data (compressed from the representations extracted by the encoder) in the raw (first row) and reconstructed (second row) real testing holograms using TSNE. Two images with (a) show the results from the encoder trained in experiment P; two images with (b) show the results from the encoder trained in experiment C1; two images with (c) show the results from the encoder trained in experiment C2; and two images with (d) show the results from the encoder trained in experiment C3.

Table 6. Clustering results from experiment P and experiments C1–C3 respectively, based on F1 score and accuracy when used to cluster the real testing holograms (Group 2).

	Shape	Experiment P (transfer learning)		Experiment C1 (transfer learning)		Experiment C2		Experiment C3	
		F1 Score	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy
Raw Holograms	circle	0.614	75.9%	0.601	76.2%	0.136	64.1%	0.274	47.1%
	triangle	0.615		0.605		0.560		0.409	
	rectangle	0.917		0.926		0.891		0.646	
	diamond	0.729		0.737		0.549		0.351	
Reconstructed Holograms	circle	0.382	68.1%	0.414	67.7%	0.271	70.2%	0.216	58.4%
	triangle	0.702		0.526		0.767		0.538	
	rectangle	0.947		0.912		0.950		0.868	
	diamond	0.429		0.596		0.568		0.342	

Note: Each value is the mean of the results from five experiments.

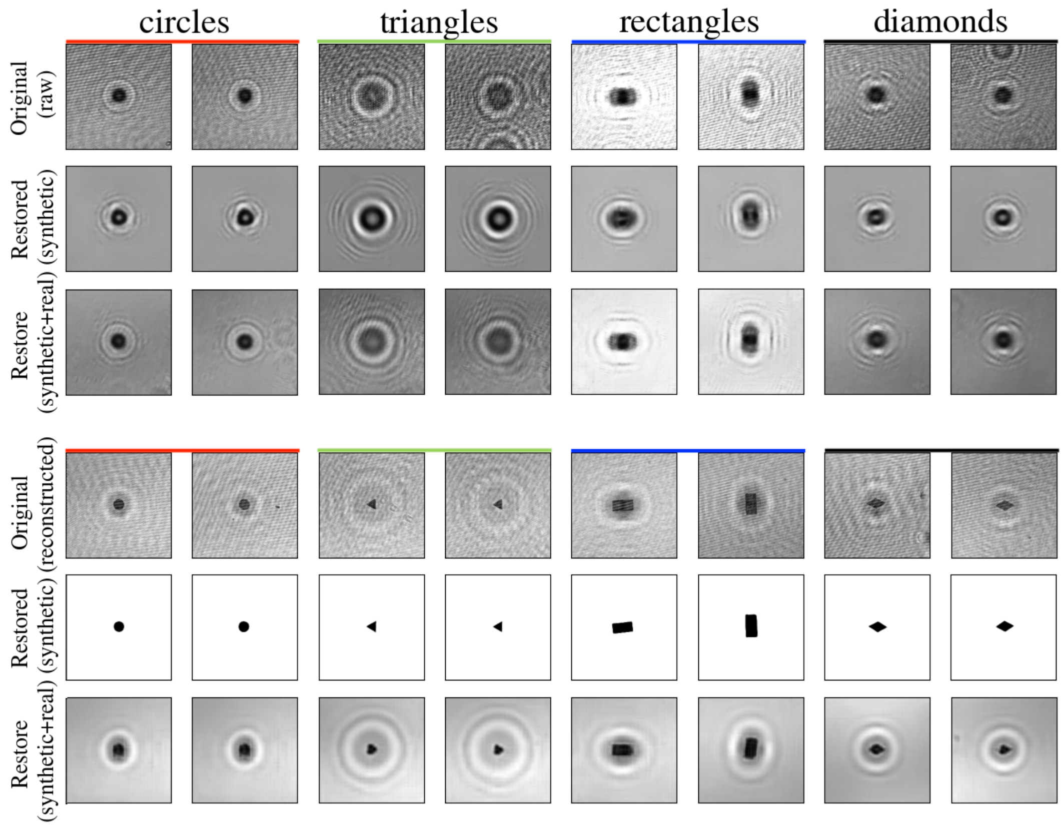

Fig. 12. Two output images in each shape from the autoencoders trained only on the synthetic data, and synthetic and real data respectively. The first three rows show the results of raw holograms, the bottom three rows show the results of reconstructed holograms.

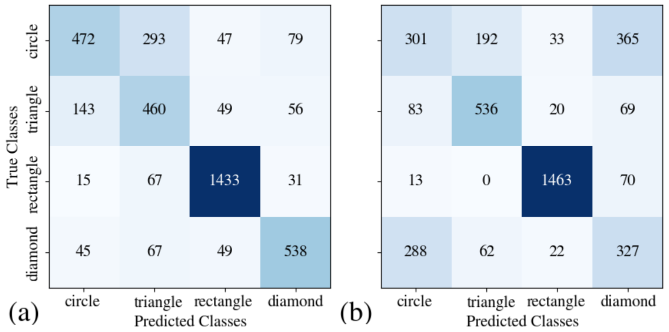

Fig. 13. Confusion matrices of the clustering results in the raw (a) and reconstructed (b) holograms in the real testing data using the models trained on both the synthetic training data and real training data (transfer learning).

changes in the reconstructed holograms.

- The SOM used is flexibly compensating and it can result in a good clustering accuracy, though representations are not well extracted.

Funding. NERC-JST SICORP Marine Sensor Proof of Concept under project code NE/R01227X/1

Acknowledgments. This work is funded by a joint UK-Japan research program (NERC-JST SICORP Marine Sensor Proof of Concept under project code NE/R01227X/1).

Disclosures. The authors declare no conflicts of interest.

Data availability. Data underlying the results presented in this paper are not publicly available at this time but may be obtained from the authors upon reasonable request.

Supplemental document. See the Supplemental Document for supporting content.

REFERENCES

1. T. Kreis, "Application of digital holography for nondestructive testing and metrology: a review," IEEE T. Ind. Inform. **12**(1), 240–247 (2016).
2. H. Sun, P. W. Benzie, N. Burns, D. C. Hendry, M. A. Player, and J. Watson, "Underwater digital holography for studies of marine plankton," Philos. Trans. R. Soc. A **366**(1871), 1789–1806 (2008).
3. G. Graham, and W. Nimmo-Smith, "The application of holography to the analysis of size and settling velocity of suspended cohesive sediments," Limnol. Oceanogr.: Methods **8**, 1–15 (2010).
4. A. Bochdansky, M. Jericho, and G. Herndl, "Development and deployment of a point-source digital inline holographic microscope for the study of plankton and particles to a depth of 6000 m," Limnol. Oceanogr.: Methods **11**, 28–40 (2013).
5. H. Sun, B. Song, H. Dong, B. Reid, M. A. Player, J. Watson, and M. Zhao, "Visualization of fast-moving cells in vivo using digital holographic video microscopy," J. Biomed. Opt. **13**(1), 014007 (2008).
6. Y. N. Nygate, M. Levi, S. K. Mirsky, N. A. Turko, M. Rubin, I. Barnea, G. Dardikman-Yoffe, M. Haifler, A. Shalev, and N. T. Shaked, "Holographic virtual staining of individual biological cells," PNAS **117**(17), 9223–9231 (2020).
7. C. Cheng, W. Hwang, C. Chen, and X. Lai, "Efficient FPGA-based fresnel transform architecture for digital holography," J. Disp. Technol. **10**(4), 272–281 (2014).
8. H. Chen, W. Hwang, C. Cheng, and X. Lai, "An FPGA-based autofocusing hardware architecture for digital holography," IEEE T. Comput. Imag. **5**(2), 287–300 (2019).
9. O. Backoach, S. Kariv, P. Girshovitz, and N. T. Shaked, "Fast phase processing in off-axis holography by CUDA including parallel phase unwrapping," Opt. Express **24**(4), 3177–3188 (2016).
10. T. Pitkäaho, A. Manninen, and T. J. Naughton, "Performance of autofocus capability of deep convolutional neural networks in digital holographic microscopy," in *Digital Holography and Three-Dimensional Imaging (OSA 2017),* paper W2A.5.
11. Z. Ren, Z. Xu, and E. Y. Lam, "Learning-based nonparametric autofocusing for digital holography," Optica **5**(4), 337–344 (2018).
12. Y. Wu, Y. Rivenson, Y. Zhang, Z. Wei, H. Günaydin, X. Lin, and A. Ozcan, "Extended depth-of-field in holographic imaging using deep-learning-based autofocusing and phase recovery," Optica **5**(6), 704–710 (2018).
13. P. Baldi, "Autoencoders, Unsupervised Learning, and Deep Architectures," in *Proceedings of Machine Learning Research (2012),* pp. 37–49.
14. G. Dong, G. Liao, H. Liu, and G. Kuang, "A review of the autoencoder and its variants: A comparative perspective from target recognition in synthetic-aperture radar images," IEEE Geosc. Rem. Sen. M. **6**(3), 44–68 (2018).
15. J. Sun, Q. Chen, Y. Zhang, and C. Zuo, "Optimal principal component analysis-based numerical phase aberration compensation method for digital holography," Opt. Lett. **41**(6):1293-1296 (2016).
16. S. J. Wetzel, "Unsupervised learning of phase transitions: From principal component analysis to variational autoencoders," Phys. Rev. **E96**, 022140 (2017).
17. C. Xing, L. Ma, and X. Yang, "Stacked denoise autoencoder based feature extraction and classification for hyperspectral images," J. Sensors **2016**, 3632943 (2016).
18. P. Liang, W. Shi, and X. Zhang, "Remote sensing image classification based on stacked denoising autoencoder," Remote Sens. **10**(1), 16 (2018).
19. T. Yamada, A. Prügel-Bennett and B. Thornton, "Learning features from georeferenced seafloor imagery with location guided autoencoders," J. Field Robotics **38**(1), 52–67 (2021).
20. F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," Proc. IEEE **109**(1), 43–76 (2021).
21. P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning (2008),* pp. 1096–1103.
22. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM **60**(6), 84–90 (2017).
23. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F.-F. Li, "ImageNet Large Scale Visual Recognition Challenge," Int. J. Comput. Vis. **115**, 211–252 (2015).
24. A. Tewari, M. Zollhöfer, F. Bernard, P. Garrido, H. Kim, P. Pérez, and C. Theobalt, "High-fidelity monocular face reconstruction based on an unsupervised model-based face autoencoder," IEEE T. Pattern Anal. Machine Intell. **42**(2), 357–370 (2020).
25. S. H. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, "Impact of fully connected layers on performance of convolutional neural networks for image classification," Neurocomputing **378**, 112–119 (2020).
26. H. Lee, J. Kim, B. Kim, and S. Kim, "Convolutional autoencoder based feature extraction in Radar data analysis," in *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS) (2018),* pp. 81–84.
27. M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision – ECCV 2014,* D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds. (Springer, 2014), pp. 818–833.
28. D. Dumitrescu and C.-A. Boiangiu, "A study of image upsampling and downsampling filters," Computers **8**(2), 30 (2019).
29. U. Schnars, C. Falldorf, J. Watson, and W. Jüptner, "Digital Holography," in *Digital Holography and Wavefront Sensing,* 2nd ed. (Springer, 2015),

ch. 2, pp. 39–68.

30. T. Kohonen, “Essentials of the self-organizing map,” Neural Networks **37**, 52–65 (2013).

31. S. A. Mingoti and J. O. Lima, “Comparing SOM neural network with Fuzzy c-means, K-means and traditional hierarchical clustering algorithms,” Eur. J. Oper. Res. **174**(3), 1742–1759 (2006).

32. T. Latychevskaia and H. Fink, “Practical algorithms for simulation and reconstruction of digital in-line holograms,” Appl. Opt. **54**(9), 2424–2434 (2015).

33. D. M. W. Powers, “Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation,” J. Mach. Learn. Technol. **2**(1), 37–63 (2011).

34. A. Tharwat, “Classification assessment methods,” Appl. Comput. Inform., ahead-of-print (2020).

35. N. M. Burns and J. Watson, “Robust particle outline extraction and its application to digital in-line holograms of marine organisms,” Opt. Eng., **53**(11), 112212 (2014).

36. M. Mortensen, L. Dalcin, and D. E. Keyes, “mpi4py-fft: Parallel Fast Fourier Transforms with MPI for Python,” JOSS **4**(36), 1340 (2019).

37. “multiprocessing — Process-based parallelism,” https://docs.python.org/3/library/multiprocessing.html. (accessed on Jul. 22, 2021).

38. “TensorFlow,” https://www.tensorflow.org. (accessed on Jul. 22, 2021).

39. G. Vettigli, “MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map,” https://github.com/JustGlowing/minisom. (accessed on Jan. 13, 2021).

40. L.v.d. Maaten and G. Hinton, “Visualizing data using t-SNE,” J. Mach. Learn. Res. **9**, 2579–2605 (2008).

41. W. S. Noble, “What is a support vector machine?,” Nat. Biotechnol. **24**, 1565—1567 (2006).