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*

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 explore the possibility of performing object clustering on holograms that have not been reconstructed, *i.e.* images of raw interference patterns, using the latent representations of a deep-learning autoencoder and a self-organising mapping network in a fully unsupervised manner. We demonstrate this concept on synthetically generated holograms of different shapes, where clustering of raw holograms achieves an accuracy of 94.4%. This is comparable to the 97.4% accuracy achieved using the reconstructed holograms of the same targets. Directly clustering raw holograms takes less than 0.1 second per image using a low-power CPU board. This represents a three-order of magnitude reduction in processing time compared to clustering of reconstructed holograms, and makes it possible to interpret targets in real time on low-power sensor platforms. Experiments on real holograms demonstrate significant gains in clustering accuracy through the use of synthetic holograms to train models. Clustering accuracy increased from 47.1% when the models were trained only on the real raw holograms, to 64.1% when the models were entirely trained on the synthetic raw holograms, and further increased to 75.9% when models were trained on the both synthetic and real datasets using transfer learning. These results are broadly comparable to those achieved when reconstructed holograms are used, where the highest accuracy of ~70% achieved when clustering raw holograms outperforms the highest accuracy achieved when clustering reconstructed holograms by a significant margin for our datasets.

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 micro-particles [2–4] and biological cells *in vivo* [5, 6]. Since raw holograms consist of the interference patterns generated when particles are in the path of coherent light, it is normally necessary to first reconstruct holograms at specific distances (the focused reconstructions) to extract the shapes of the particles before further analysis, *e.g.* object classification or size analysis, can be performed. However, hologram reconstruction is a computationally intensive process. It becomes more expensive when the distance to the target is not known 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 motivates 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 feature extraction methods, *e.g.* principal component analysis (PCA) [15], autoencoders can model more complex, nonlinear relationships between inputs and their extracted features, *i.e.* latent representations [16]. This flexibly makes autoencoders effective at learning features from datasets gathered under different conditions or when different instruments are used. The latent representations extracted by autoencoders can be used for clustering without the need for any human supervision. This has been effectively demonstrated for various types of optical image [17–20]. However, there have been no previous studies investigating their use for clustering of raw digital holograms.

In this paper, we investigate how to learn features from raw holograms and cluster holograms based on these features using an end-to-end unsupervised workflow. 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 features, which can be challenging to obtain in applications where targets of interest are sparse and have unbalanced class distributions (*e.g.* marine micro-particle imaging). Therefore, we investigate how to improve the efficiency of training unsupervised models using synthetically generated data. The concept of directly interpreting raw holograms is first demonstrated entirely using synthetic holographic data. Next, we explore methods to analyse real holograms using transfer learning [21], where models are first pre-trained on synthetic holograms before training on a small number of real holograms. The performance of this proposed method is compared to alternative transfer learning methods that use generic image databases, and methods that use only synthetic holograms for training. The proposed workflow is demonstrated on a low power CPU board to show its practical use for *in situ* applications.

2. FEATURE LEARNING USING 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] an image, x_r, that is as close to the initial input image x as possible. Typically, noise is added to inputs so that the encoder learns to denoise images and extract more robust representations of the original inputs [14, 22].

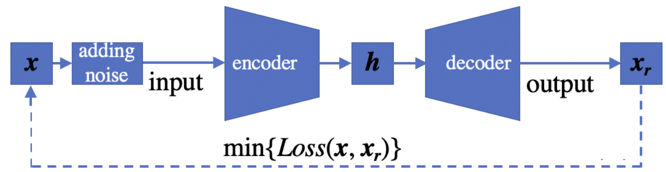

Fig. 1. Flowchart of an autoencoder with denoising. x, h and x_r signify an input image, its latent representation and restored image respectively. $Loss(x, x_r)$ is a loss function which calculates the error between x and x_r.

The model learns by minimising the difference, or loss, between x and x_r for all the images in a training dataset. The process can be described as follows:

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

where φ and ϕ are the mappings 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 h that have reduced dimensions compared to x and x_r, and can be used as features for unsupervised clustering or supervised classification of the inputs.

The autoencoder used in this work is based on the AlexNet architecture [23, 24]. 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 are each followed by max pooling layers) and 3 fully-connected layers. The relatively simple architecture compared to more recent convolutional neural networks (CNNs) makes it suitable for use in autoencoders, as demonstrated in [19, 20, 25].

In this work, two modifications are made to the original AlexNet architecture as described in Section 1 of the supplementary 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 architecture are useful for solving highly complex classification problems [26]. However, these fully-connected layers comprise 94% of the parameters in AlexNet and allow geometric structures in the input images to be lost in the extracted features [27]. In contrast, the convolutional layers preserve spatial locality [23]. 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 (each followed by a max pooling layer). This modification efficiently preserves geometric characteristics in the extracted features, which improves the learning efficiency for spatially structured images like raw holograms, and speeds up the training process with a reduced network size. Details of these improvements are described in Section 3-B of the supplementary document).

The first modified convolutional layer uses 96 filters with a kernel size of 3 × 3 and scanning stride of 1 × 1. The same padding strategy [28] is used in this layer, which results in this

[1] In this paper the output of the decoder is called restoration (image generated from the latent representation); the output of the hologram reconstruction algorithm is called reconstruction (focused image that has been generated from the raw interference patterns).

layer outputting a datum of size 6 × 6 × 96. After max pooling with a pooling size of 3 × 3 and a scanning stride of 3 × 3, the output datum size becomes 2 × 2 × 96. The second convolutional layer controls the number of the latent features. Its output size is 2 × 2 × 40. ReLU (rectified linear unit) activation functions are used in these two convolutional layers. After max pooling, a 40-dimension latent representation is obtained from each input image. 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 supplementary document). Its decoder is mirror-symmetrical, where convolutional layers are transposed to transconvolutional layers [29], and the max pooling layers are transposed to upsampling layers [30].

To address the background noise that exits in holograms [31], a denoising step is added to the autoencoder to reduce the effect of noise on feature extraction (Section 3-C in the supplementary document). The training parameters for the autoencoder are described in Section 2-A of the supplementary document.

3. CLUSTERING MODEL

In this work, objects are clustered using a self-organising mapping (SOM) network [32]. The SOM is a well-established unsupervised learning model that is built using a pre-defined 2-D net of neurons [33]. Unlike the error-correction-based learning in other networks (*e.g.* gradient descent in backpropagation), competitive learning [32] 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 are mapped to the same clusters. Further details of the SOM used in this work can be found in Section 2-B of the supplementary document.

4. DATASETS

For applications such as marine micro-particle imaging, it can be difficult to prepare large datasets of real holographic imagery for training a deep-learning autoencoder. A possible solution is to generate synthetic holograms and use these to train a model. The trained model can be used directly, or used as pre-trained outputs to initialise further training using a small dataset of real holographic images (*i.e.* transfer learning). Artificial noise is added to the synthetic holograms to facilitates the denoising training process in the autoencoder.

Experiments are performed on both the interference patterns of raw holograms and equivalent reconstructed images of four simple geometries: circle, triangle, rectangle, and diamond. Real holograms are obtained using a 200 × 200 mm^2 glass plate with these shape patterns etched on it. The diameter of the circle and the smallest edge of other patterns is 100 µm, as shown in Fig. 2, where the etched shapes on the plate have 1 mm separation between them. The synthetic dataset is generated using the same shapes without any neighbours.

Real dataset: An in-line holographic camera was used to take holograms of the shape plate. The setup is shown in Fig. 3 and is based on a previously described system [34]. A 532 nm, single-longitudinal mode continuous wave laser (Elforlight) is used as the light source. The beam intensity is controlled using a variable neutral density filter, while a spatial filter (items ③ and ④ provide a spatially coherent and uniform beam. This beam is collimated using a lens ⑤ before illuminating a complementary

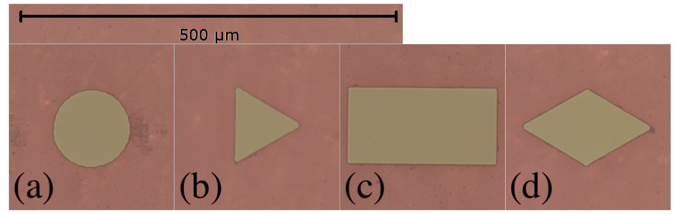

Fig. 2. Microscopic photographs of four shapes. (a) – circle; (b) – triangle; (c) – rectangle; (d) – diamond.

metal-oxide-semiconductor (CMOS) image sensor (JAI GO-5100-USB, ⑥) that has a resolution of 2464 × 2056, pixel pitch of 3.45 µm and an active detection area of 8.5 mm × 7.09 mm.

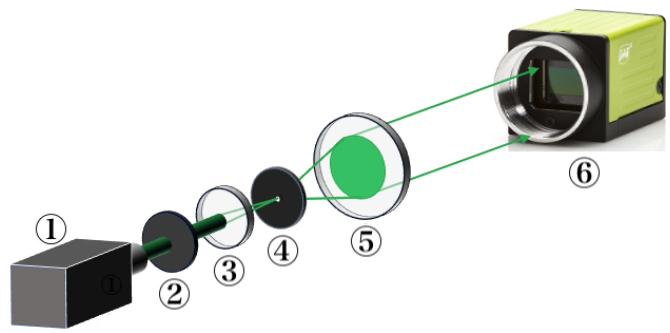

Fig. 3. Schematic diagram of the in-line structure hologram recorder used in this work. ① – laser, ② – neutral density filter, ③ – microscopic objective lens, ④ – pinhole, ⑤ – collimating convex lens, ⑥ – CMOS image sensor.

The shape plate is placed in the laser beam path, between the collimating lens ⑤ and sensor ⑥. The dataset consists of holograms where the distance of the plate from the sensor varies between 10 mm to 60 mm, and different sensor exposure times (10, 40, 70, 100, 130, 160, 190 and 220 µs) and relative plate orientations (between -90° and 90°) are used. Fig. 4 shows four holograms of rectangles recorded under different conditions.

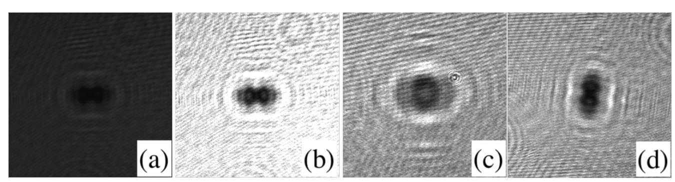

Fig. 4. Four hologram samples of a rectangle under different conditions. (a) recorded at 17.90 mm with 10 µs exposure time; (b) recorded at 17.90 mm with 220 µs exposure time; (c) recorded at 47.70 mm with 130 µs exposure time; (d) recorded at 17.85 mm with 130 µs exposure time and close to 90° rotation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

Two independent sets of real holographic data are used. One set (Group 1) is used for autoencoder training, and the other (Group 2) is used to test the trained models. Each hologram is cropped to 300 × 300 pixels around the target (as discussed in Section 3-D of the supplementary document), resulting in 4,180 cropped holograms in Group 1 and 3,844 in Group 2 (see Table 1). These are reconstructed using the angular spectrum method [35], with examples of reconstructed holograms shown in Fig. 5.

Table 1. Number of real holograms for each shape.

Group	Circle	Triangle	Rectangle	Diamond	Total
1	780	887	1522	991	4,180
2	891	708	1546	699	3,844

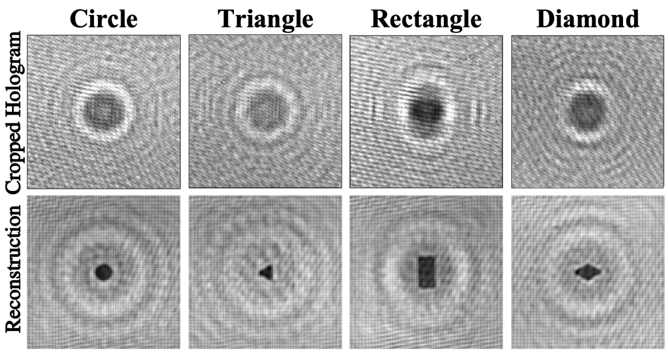

Fig. 5. Cropped holograms of four shapes with the size of 300 × 300 and their reconstructions.

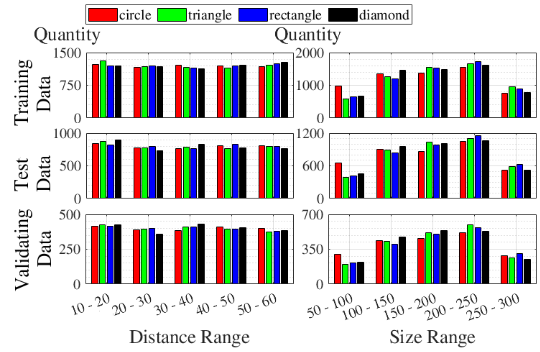

Fig. 6. Histogram of recording distances and shapes' sizes in three groups.

Synthetic dataset: Raw holograms are simulated from the target shapes using the angular spectrum method. The parameters used for the simulation are shown in Table 2. The size and recording distance of the shapes are randomly selected from within the given ranges. The centre and orientation of the shapes are also randomly chosen, but are restricted so that the shape is fully shown within the boundary of the image.

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 are created: training data consisting of 24,000 holograms, validating data with 8,000 holograms and test data with 16,000 holograms. In each group, the number of each shape is equal. Histograms of the recording distances and shapes' sizes for each group are shown in Fig. 6. Regarding the recording distance, the number of the holograms of each shape in each range is similar. Most holograms lie within the size range of 100 – 250 μm, which matches the sizes of the shapes in the glass plate (see Fig. 2).

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

Noise is added to the synthetic holograms by taking real holograms without any targets and superimposing randomly cropped regions of them as background noise in the synthetic holograms (see Fig. 8).

5. RESULTS AND ANALYSIS

The clustering performance of the proposed method was verified on the raw holograms of the entirely synthetic, entirely real and on combined synthetic and real hologram datasets using transfer learning. The results were compared to the equivalent performance with reconstructed holograms for all the conditions investigated in this work. In the first set of experiments, both training and evaluation were performed on the synthetic data to validate our concept. Next, experiments were performed to cluster the real dataset. For the transfer learning experiments, the autoencoder was pre-trained on the synthetic holograms and fine-tuned using the real hologram data in Group 1. Afterwards, the encoder was used to extract the latent representations from the corresponding real hologram data in Group 2. These representations were used as features to cluster the real holograms in an unsupervised manner. For comparison, we also performed transfer learning using the generic ImageNet database [24] for pre-training. Besides this, only synthetic data and only real data from Group 1 were respectively used for training the models. These experimental conditions are shown in Table 3.

The clustering performance was assessed using the overall accuracy and F1 score [36, 37] 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 can be directly integrated into a compact digital holographic microscope for use *in situ*.

All the algorithms in this work were implemented in Python programming language. The angular spectrum algorithm [35] was used to reconstruct a hologram at a given distance, and the autofocusing method described in [38] was used to automatically detect the focused reconstruction across the entire recording distance range. In order to speed up the algorithms of angular spectrum and autofocusing, two Python-based modules were used: **mpi4py-fft** [39] for parallel computing the fast Fourier

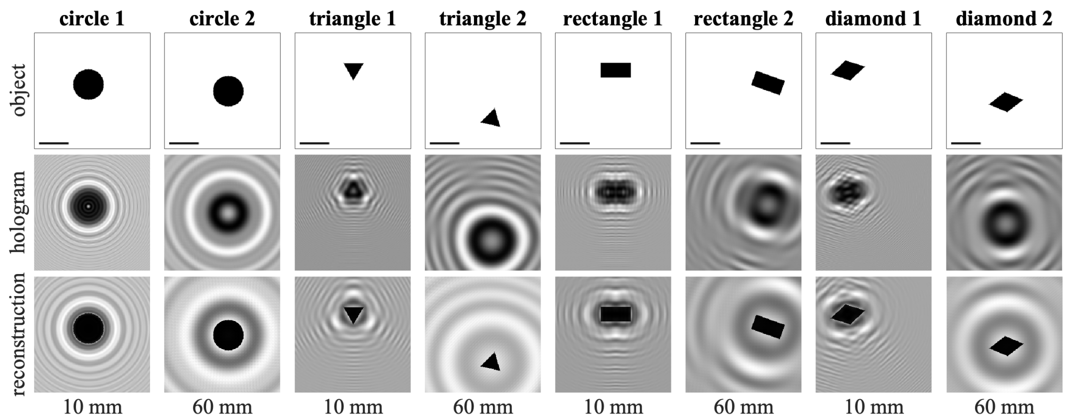

Fig. 7. Two examples of each shape, including original shapes (in the first row), corresponding synthetic holograms (in the second row) and their reconstructions (in the third row). Number below each column gives the recording distance of the hologram. The scale lines in the first row indicate 200 μm.

Table 3. Description of four sets of experiments.

	Experiment	data for training autoencoder	data for training SOM	test 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.

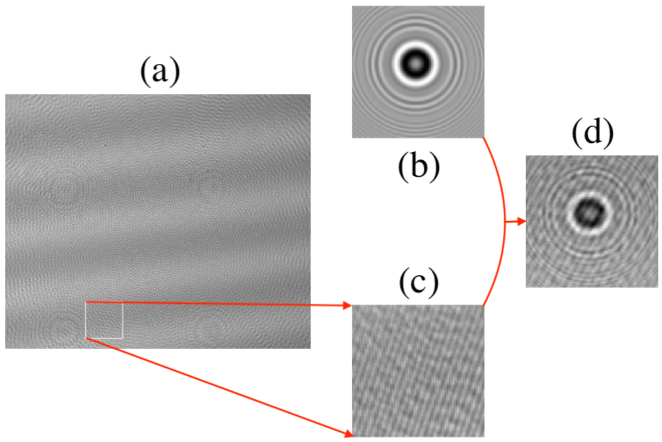

Fig. 8. An example of adding noise to the synthetic hologram. The noise image (c) is cropped from a background hologram (a), and it is added to a synthetic hologram (b) to create the final synthetic hologram (d).

transforms used by the algorithms, and **multiprocessing** [40] for parallel execution of reconstruction across the recording distance range. The autoencoder was developed, trained and tested using **Tensorflow** [41]. The SOM model was built, trained and tested using the open-source library of **MiniSom** [42].

A. Feature learning and clustering of 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) for 100 epochs. The fact that the loss is similar for training and validation indicates that the model is able to generalise, without over-fitting the synthetic data. The result also shows that convergence is achieved after ~40 epochs.

Fig. 10 shows the TSNE (t-distributed stochastic neighbour embedding) [43] plots of the latent representations extracted from the raw and reconstructed holograms in the test data by the corresponding trained encoders. It shows that there is larger separation between the different shapes in the reconstructed holograms, with some merging between different shapes occurs in the plot of the raw data. Some raw holograms of circles are mixed with the triangle cluster, and this is reflected in the clustering scores of these shapes being lower in the raw holograms than the reconstructed holograms.

The autoencoder and SOM were trained five times, and each trained encoder and SOM pair were used to cluster the corresponding raw and reconstructed synthetic 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 require labelled training data of the target shapes, where in this case the ground truth synthetic data was available. The first method used a support vector machine (SVM) [44] that was trained on the features extracted from the training data by the encoder. The trained SVM was then used to classify the test data (training parameters are

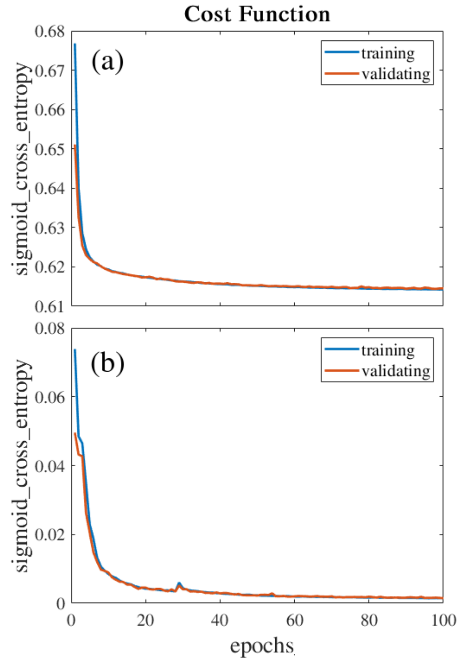

Fig. 9. Loss curves for autoencoder training and validation on the raw (a) and reconstructed (b) synthetic holograms. Each loss value is the mean of the results from five experiments.

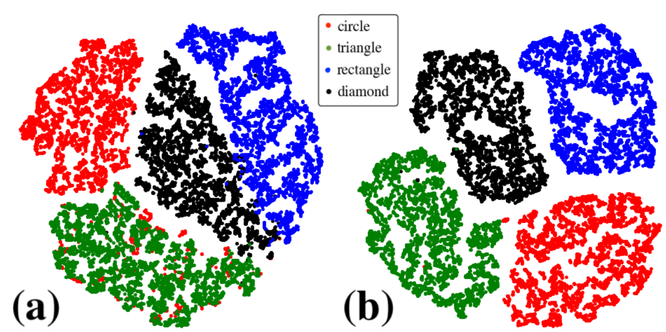

Fig. 10. TSNE visualisation of the latent representation space extracted by the encoder for (a) the raw and (b) the reconstructed synthetic hologram test data.

given in Section 2-C of the supplementary document). The second method used AlexNet[1] to directly classify the input images based on the labelled training data. Table 4 shows their performance for the raw and reconstructed holograms in the test data. Clustering using the proposed unsupervised method - SOM - achieves a high accuracy of 94.4% and 97.3% for the raw and reconstructed holograms, respectively. The corresponding F1 scores for each target shape are lower for the raw holograms than the reconstructed holograms. The two supervised classifiers achieve higher accuracy scores, which is expected since labelled training data is provided to the classifiers. The results show that a high level of accuracy can be achieved when directly analysing raw holograms that is comparable to processing reconstructed holograms. This has significant implications for *in situ* applications as it avoids the large computational overhead needed to reconstruct holograms. 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 5 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 to reflect a realistic *in situ* operating scenario. The time required to train the autoencoder and SOM are almost identical for the raw and reconstructed holograms. The biggest cost is in the reconstruction of the holograms, which takes more than 13 times the combined training time. This processing step is not needed when interpreting raw holograms directly. Clustering the entire test dataset of 16,000 images using each trained encoder and SOM pair takes approximately 1,500 s, *i.e.* and average of ~0.09 s to process each hologram input. This would allow real-time clustering on the lower-power CPU board for an image acquisition rate of up to 10 Hz. However, reconstructing each hologram on the same lower-power CPU board takes ~14 s per image, which forms significant bottleneck for real-time clustering of reconstructed holograms. It should be noted that hardware optimisation, such as the use of Field Programmable Gate Array (FPGA) or Graphics Processing Unit (GPU) embedded single board computers can allow real-time reconstruction at faster rates [7–9]. However, this comes at the cost of higher power consumption, which is not ideal for long term monitoring applications where low-power electronics solutions are required.

B. Feature learning and clustering of real holograms

In our proposed method (P), experiments were performed where the autoencoder was first pre-trained on the synthetic training holograms and then fine-tuned using a small set of real holograms (Group 1) using transfer learning (see Section 2-D in the supplementary document). Similarly, the SOM pre-trained on the synthetic training data was also fine-tuned using the features of the holograms in Group 1 extracted by the fine-tuned autoencoder. The fine-tuned encoder and SOM were then used to extract and cluster the latent representations from real holograms in the test dataset in Group 2. The real holograms for testing were fed to the fine-tuned autoencoder and SOM. For comparison, three other sets of experiments were carried out with the real test holograms: C1. the autoencoder was pre-trained on the ImageNet dataset[1] (2012 [24]) and fine-tuned on the real holographic training data; the SOM was trained on the real training data based on the features extracted by the trained encoder; C2. 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. These experiment conditions are summerised in Table 3. The parameters for fine-tuning the autoencoder and SOM are kept the same as those used in pre-training (see Section 2-A&B in the supplementary document).

The latent representations of the real test holograms extracted by the encoders from these four experiments are visualised in Fig. 11, and the results of clustering are shown in Table 6. Compared to the TSNE plots of the synthetic data in Fig. 10, the latent representations have decreased separation between the points of different shapes, where this is likely due to the unmodelled complexities of recording real holograms compared to generating synthetic ones. The TSNE visualisation for the proposed transfer learning method P (encoder pre-trained on synthetic data and fine-tuned on real data) shows good separation between the different shape classes for the raw holograms (Fig. 11-(a-1)), with the rectangles clearly separated from other shapes. However, there is some mixing between the circle, triangle and diamond classes, which we expect to degrade the clustering performance for these shapes. A similar trend is seen for the conditions C1 (Fig. 11-(b-1))and C2 (Fig. 11-(c-1)). For C3 (Fig. 11-(d-1)), which was trained only on the real holograms, we see a higher level of mixing overall, and so we expect poor clustering performance. This result shows that the use of synthetic data and transfer learning can benefit feature extraction. A possible explanation for the better performance is that availability of a large number of training examples, where there are only 24,000 images in the synthetic training data, and 1,281,167 images in the ImageNet 2012 training dataset, both of which are larger than the 4,180 real holograms available for training in Group 1.

For the reconstructed holograms, although there is some change in the TSNE for conditions P (Fig. 11-(a-2)), C1 (Fig. 11-(b-2)) and C2 (Fig. 11-(c-2)), the degree of mixing and separation are similar to the raw holograms. For condition C3 (Fig. 11-(d-2)) however, there is a clear reduction in mixing and improvement in the separation of the rectangles. For conditions P, C1, and C2, the TSNE distributions for the raw holograms and reconstructed holograms have different appearances but it is not clear if either has a clear advantage over the other. This is favourable as we do not expect a large differences in performance when interpreting the raw holograms, which has a far lower computational overhead than the reconstructed holograms.

Regarding the differences between P (synthetically pre-trained and fine-tuned using real data) and C2 (synthetic data trained encoder), some insight can be drawn from Fig. 12, which shows two output images of each shape restored by the autoencoders for conditions P and C2 respectively. The autoencoder trained only on the synthetic reconstructed data with denoising allows it to restore reconstructed holograms with clear shape outlines, but fine-tuning the model on the real reconstructed holograms reduces this capability, as seen by the distortion of the diamond shape and rounding of the triangles. For raw holograms however, the fine-tuned autoencoder retains the distinctive characteristics of the original inputs better than the autoencoder trained only on the synthetic holograms. This is noticeable

[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.

[1] The images were converted into grayscale.

Table 4. Results of the three methods based on the F1 scores and accuracy when used to cluster/classify the synthetic holograms in the test dataset.

	Shape	Encoder+SOM		Encoder+SVM		AlexNet	
		F1 Score	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy
Raw Holograms	Circle	0.933 (±0.009[a])	94.4% (±0.4%)	0.980 (±0.007)	98.9% (±0.3%)	1.000 (±0)	99.8% (±0.1%)
	Triangle	0.930 (±0.006)		0.980 (±0.004)		1.000 (±0)	
	Rectangle	0.966 (±0.009)		1.000 (±0)		1.000 (±0)	
	Diamond	0.948 (±0.013)		1.000 (±0)		1.000 (±0)	
Reconstructed Holograms	Circle	0.975 (±0.014)	97.4% (±1.5%)	1.000 (±0)	99.9% (±0.1%)	1.000 (±0)	100.0% (±0%)
	Triangle	0.978 (±0.013)		1.000 (±0)		1.000 (±0)	
	Rectangle	0.980 (±0.017)		1.000 (±0)		1.000 (±0)	
	Diamond	0.962 (±0.025)		1.000 (±0)		1.000 (±0)	

[a] standard deviation.
Note: Each value is the mean of the results from five experiments.

Table 5. Processing time for the models used to extract features from and cluster the raw and reconstructed holograms.The times shown are for processing the entire training and test datasets, where training is performed on a high-preformance workstation, and testing is performed on a low-power single board CPU.

	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.

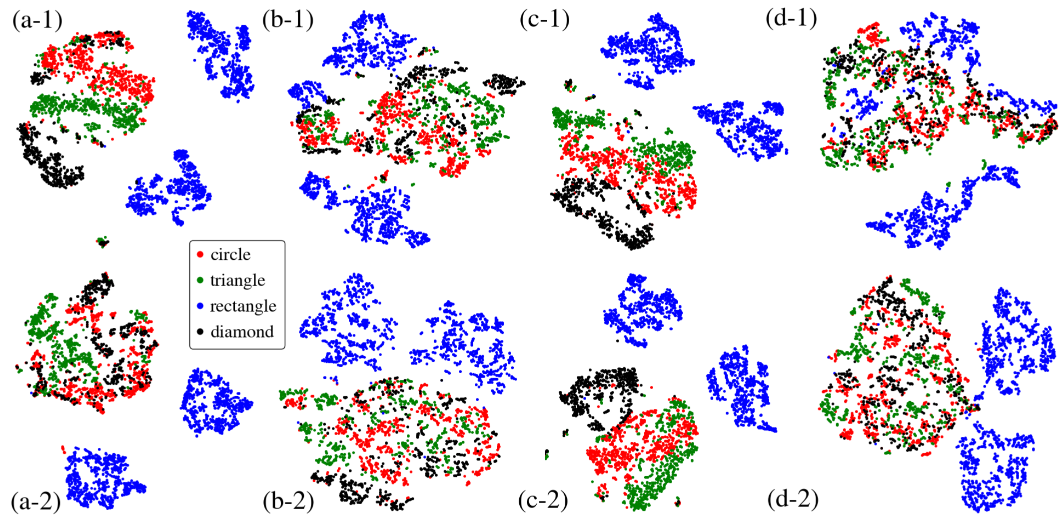

Fig. 11. TSNE visualisations of the latent representations extracted by the encoder for raw (first row) and reconstructed (second row) real test holograms. (a) shows the results for the encoder trained using the proposed condition P; (b) shows the results for the encoder trained using condition C1; (c) shows the results for the encoder trained using condition C2; and (d) shows the results for the encoder trained using condition C3. A description of the conditions is given in Table 3.

for the restored images of the raw circles, where the entirely synthetically trained autoencoder deforms the interference fringes, making them appear triangular.

The clustering performance of the real test holograms (Group 2) using the corresponding encoder and SOM pairs (Table 3) are shown in Table 6. The poorest performance is for the models were trained only on the real data (C3), where the raw holograms achieves the accuracy of 47.1% and the reconstructed holograms achieves 58.4%. The improved performance for the reconstructed holograms, particularly for the rectangles is expected based on the TSNE visualisation of the latent representations used for clustering. When the models were trained only on the synthetic data (C2), the accuracy increases for both raw and reconstructed holograms to 64.1% and 70.2%, respectively. In both C2 and C3, the reconstructed holograms perform better than the raw holograms. For the two transfer learning methods, the accuracy achieved for raw holograms further increases to ~76%, while the accuracy in the reconstructed holograms (~68%) is comparable to condition C2. Regarding accuracy, the models trained on the synthetic and real data (P) have similar performance with the models trained on the ImageNet and real data (C1), where the differences in the results are within the order of experiment repeatably. This is somewhat unexpected based on the TSNE plots, where the ImageNet pre-trained (C1) encoder appears to have a higher level of mixing between shapes than the synthetically pre-trained (P) encoder (see Fig. 11). A possible explanation is that the SOM used is limiting the ability to separate the shapes in P due to the merged boundaries between the shapes, and this is leading to a similar degree of confusion as the more intermixed distributions between the shapes in C1. Another unexpected result is that the accuracy in the raw holograms is higher than the reconstructed holograms after using transfer learning. This is reflected in Fig. 12, which shows that transfer learning does not facilitate the encoder to extract better representations from reconstructed holograms.

The performance across classes is not uniform based on the F1 score in each set of experiments. The rectangles are always resolved the best, and the circles resolved the worst both in the raw and reconstructed holograms. After using transfer learning, the circles and diamonds are better resolved in the raw holograms than the reconstructed holograms. The corresponding confusion matrices of the raw and reconstructed holograms from the experiments using condition P are shown in Fig. 13. In the raw holograms, there is a high degree of confusion between the circles and triangles. A possible reason for this can be seen in Fig. 11, where the restored interference fringes of the circles look similar to the triangles. In the reconstructed holograms, there is greater confusion between the circles and diamonds, which can again be seen in Fig. 11, where the restored diamonds have lost characteristic information about their shape.

6. CONCLUSIONS

Object clustering can be efficiently performed on raw holograms to achieve comparable performance to equivalent reconstructed holograms for the shapes investigated in this work. 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 using low power CPUs. 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. We demonstrate a modified CNN architecture that preserves geometric structure in the original images when extracting latent representations. 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.

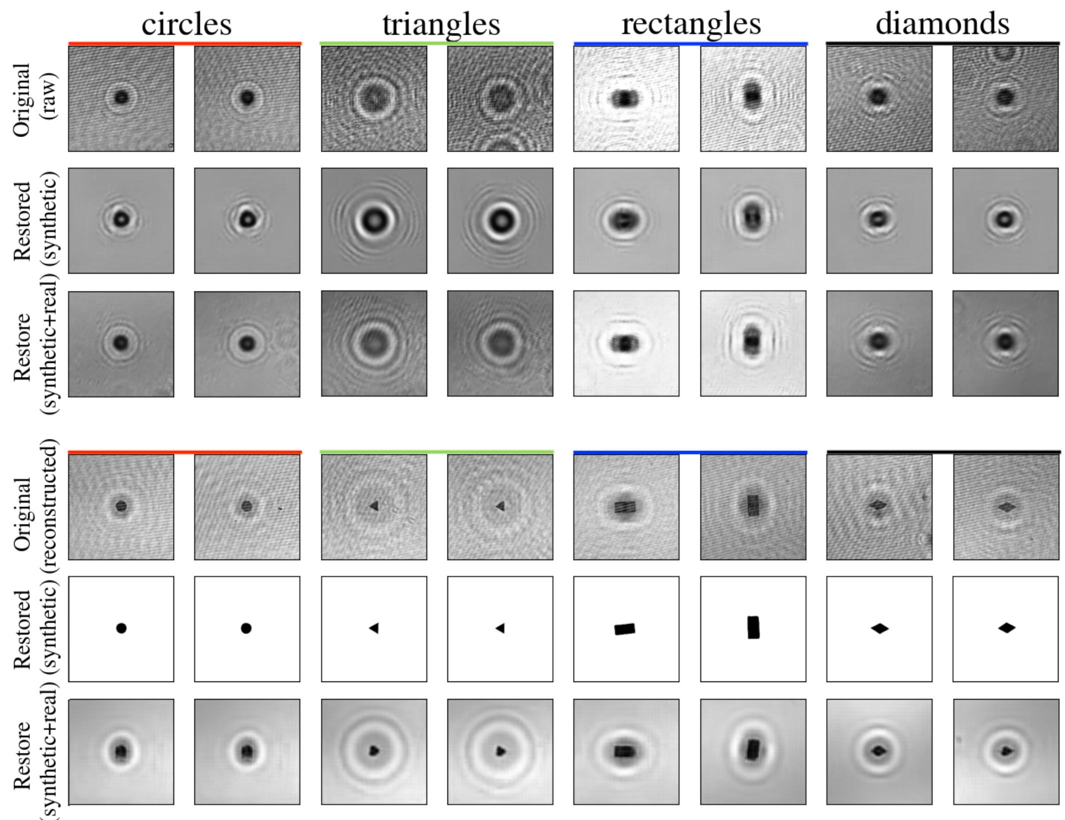

Fig. 12. Examples of input and restored output images for each shape using the autoencoders trained only on the synthetic data (C2), and synthetically pre-trained and fine-tuned with real data (P) respectively. The first three rows show the results for raw holograms, the bottom three rows show the results for reconstructed holograms.

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

	Shape	Condition P (transfer learning)		Condition C1 (transfer learning)		Condition C2		Condition C3	
		F1 Score	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy
Raw Holograms	Circle	0.614 (±0.023[a])	75.9% (±1.3%)	0.601 (±0.063)	76.2% (±2.6%)	0.136 (±0.082)	64.1% (±3.8%)	0.274 (±0.095)	47.1% (±11.9%)
	Triangle	0.615 (±0.044)		0.605 (±0.065)		0.560 (±0.022)		0.409 (±0.109)	
	Rectangle	0.917 (±0.020)		0.926 (±0.022)		0.891 (±0.061)		0.646 (±0.170)	
	Diamond	0.729 (±0.053)		0.737 (±0.044)		0.549 (±0.074)		0.351 (±0.091)	
Reconstructed Holograms	Circle	0.382 (±0.030)	68.1% (±3.0%)	0.414 (±0.044)	67.7% (±1.8%)	0.271 (±0.022)	70.2% (±7.9%)	0.216 (±0.137)	58.4% (±7.8%)
	Triangle	0.702 (±0.078)		0.526 (±0.024)		0.767 (±0.146)		0.538 (±0.102)	
	Rectangle	0.947 (±0.006)		0.912 (±0.011)		0.950 (±0.026)		0.868 (±0.100)	
	Diamond	0.429 (±0.042)		0.596 (±0.050)		0.568 (±0.083)		0.342 (±0.147)	

[a] standard deviation.
Note: Each value is the mean of the results from five experiments.

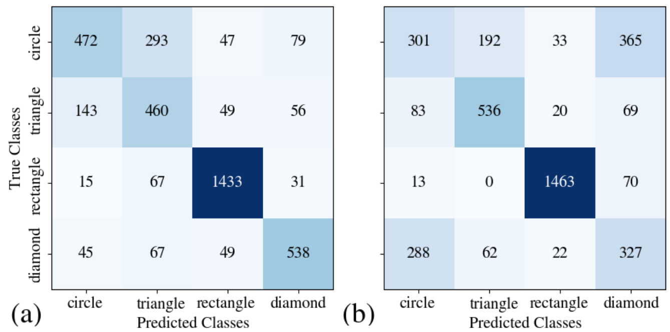

Fig. 13. Confusion matrices of the clustering results for models pre-trained on synthetic and fine-tuned on real holograms (Group 1 in Table 1) (P) for raw (a) and reconstructed (b) holograms in the real test data (Group 2 in Table 1).

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 the results for raw and reconstructed holograms, the syntheticalled trained encoders achieved 64.1% and 70.2% accuracy, respectively. This is significantly better than the results from the models trained only on the real training holograms. Further gains in performance can be achieved using transfer learning techniques, where the models are synthetically pre-trained, and then fine-tuned using real holograms. This increased the accuracy when processing raw holograms to 75.9%. Similar gains in accuracy were not however, achieved for the reconstructed holograms. This performance is comparable to the performance achieved when using a far larger (1.2 million images as opposed to 24,000 synthetic images) generic image database for pre-training, where it is suggested that the SOM used for clustering limits the final accuracy achieved by the proposed method.

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.

supplementary document. See the supplementary 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. T. Yamada, M. Massot-Campos, A. Prügel-Bennett, S. B. Williams, O. Pizarro and B. Thornton, "Leveraging Metadata in Representation Learning With Georeferenced Seafloor Imagery," IEEE Robotics and Automation Letters **6**(4), 7815–7822 (2021).
21. 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).
22. 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.
23. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM **60**(6), 84–90 (2017).
24. 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).

25. 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).
26. 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).
27. 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.
28. M. Hashemi, "Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation," J. Big Data **6**, 98 (2019).
29. 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.
30. D. Dumitrescu and C.-A. Boiangiu, "A study of image upsampling and downsampling filters," Computers **8**(2), 30 (2019).
31. 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.
32. T. Kohonen, "Essentials of the self-organizing map," Neural Networks **37**, 52–65 (2013).
33. 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).
34. Z. Liu, T. Takahashi, D. Lindsay, T. Thevar, M. Sangekar, H. Kayama-Watanabe, N. Burns, J. Watson, B. Thornton "Digital In-Line Holography for Large-Volume Analysis of Vertical Motion of Microscale Marine Plankton and Other Particles," IEEE J. Oceanic Eng. (2021).
35. T. Latychevskaia and H. Fink, "Practical algorithms for simulation and reconstruction of digital in-line holograms," Appl. Opt. **54**(9), 2424–2434 (2015).
36. 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).
37. A. Tharwat, "Classification assessment methods," Appl. Comput. Inform. **17**, 168–192 (2021).
38. 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).
39. M. Mortensen, L. Dalcin, and D. E. Keyes, "mpi4py-fft: Parallel Fast Fourier Transforms with MPI for Python," JOSS **4**(36), 1340 (2019).
40. "multiprocessing — Process-based parallelism," https://docs.python.org/3/library/multiprocessing.html. (accessed on Jul. 22, 2021).
41. "TensorFlow," https://www.tensorflow.org. (accessed on Jul. 22, 2021).
42. G. Vettigli, "MiniSom: minimalistic and NumPy-based implementation of the Self Organizing Map," https://github.com/JustGlowing/minisom. (accessed on Jan. 13, 2021).
43. L.v.d. Maaten and G. Hinton, "Visualizing data using t-SNE," J. Mach. Learn. Res. **9**, 2579–2605 (2008).
44. W. S. Noble, "What is a support vector machine?," Nat. Biotechnol. **24**, 1565—1567 (2006).