

Supplementary information for

Genetically programmable optical random neural networks

Bora Çarpınlioğlu, Uğur Teğın*

Department of Electrical and Electronics Engineering, Koç University, Istanbul, 34450, Türkiye

*uteğın@ku.edu.tr

Supplementary Discussion 1: Breast MNIST Simulation Details

Prior to conducting any experiments, we first validated our approach with beam propagation method (BPM) simulations. Since our experimental design is based on linear operations, we solve numerically the linear wave propagation equation. Working in the linear domain allows us the ability to perform one-step simulations i.e., the propagation does not need to be divided into small steps of dz .

In line with our experimental prototype, we defined a spatial grid covering the spatial light modulator (SLM) active area where modulation is performed. A Gaussian beam is created with samples from the Breast MNIST dataset encoded as phase objects. Since the samples in the dataset have small (28×28 pixels) resolution, they are upsampled to fit the active area of the SLM. Following linear beam propagation and focusing with a lens, the complex field is multiplied with one of the 4096 pre-computed random matrices, determined by genetic algorithm (GA). The random matrices are defined as follows: an initial, fully random matrix is generated corresponding to position 0. Then, the random matrix at position 1 is created with half of the elements originating from the random matrix at position 0, and the remaining half is created randomly. This procedure goes on until the last random matrix. As a result, consecutive random matrices become correlated, emulating the stepper motor rotation in the experimental setup. After multiplication with a given random matrix, normalized intensity profiles are obtained with another relaying lens and linear beam propagation, after which ridge classification is performed to obtain classification accuracy. In simulation, GA parameters of $n = 12$ generations and $p = 4$ population size are used. An example BPM script can be found on GitHub (1).

Supplementary Discussion 2: 4f Imaging System and Input Encoding

The beam expander with the Keplerian design applied to the laser beam consists of two lenses placed in parallel at a distance equal to the sum of their focal lengths and its magnification can be calculated by the ratio of the focal lengths of the constituent lenses. The lenses used in this 4f imaging system are two bi-convex lenses with focal lengths of 30 mm and 268 mm. In the 4f imaging system, input beam inversion does

not pose a problem for us because the output beam has the same intensity pattern as the input due to the azimuthal symmetry of the input Gaussian beam. As a result, SLM receives a magnified laser beam whose input intensity pattern did not change. Similarly, after reflection from the SLM, we use a 4f imaging system, again using two bi-convex lenses with focal lengths of 268 mm and 50 mm.

In the encoding step, since the size of the samples in the datasets was often smaller than the size of the SLM, we performed image upsampling using bilinear interpolation for each sample such that the illuminated region on the SLM was filled with the sample. In areas where the SLM could not be filled by the rescaled image, we simply performed zero-padding.

Supplementary Discussion 3: The Experimental Setup and the Scattering Medium Used in Experiments

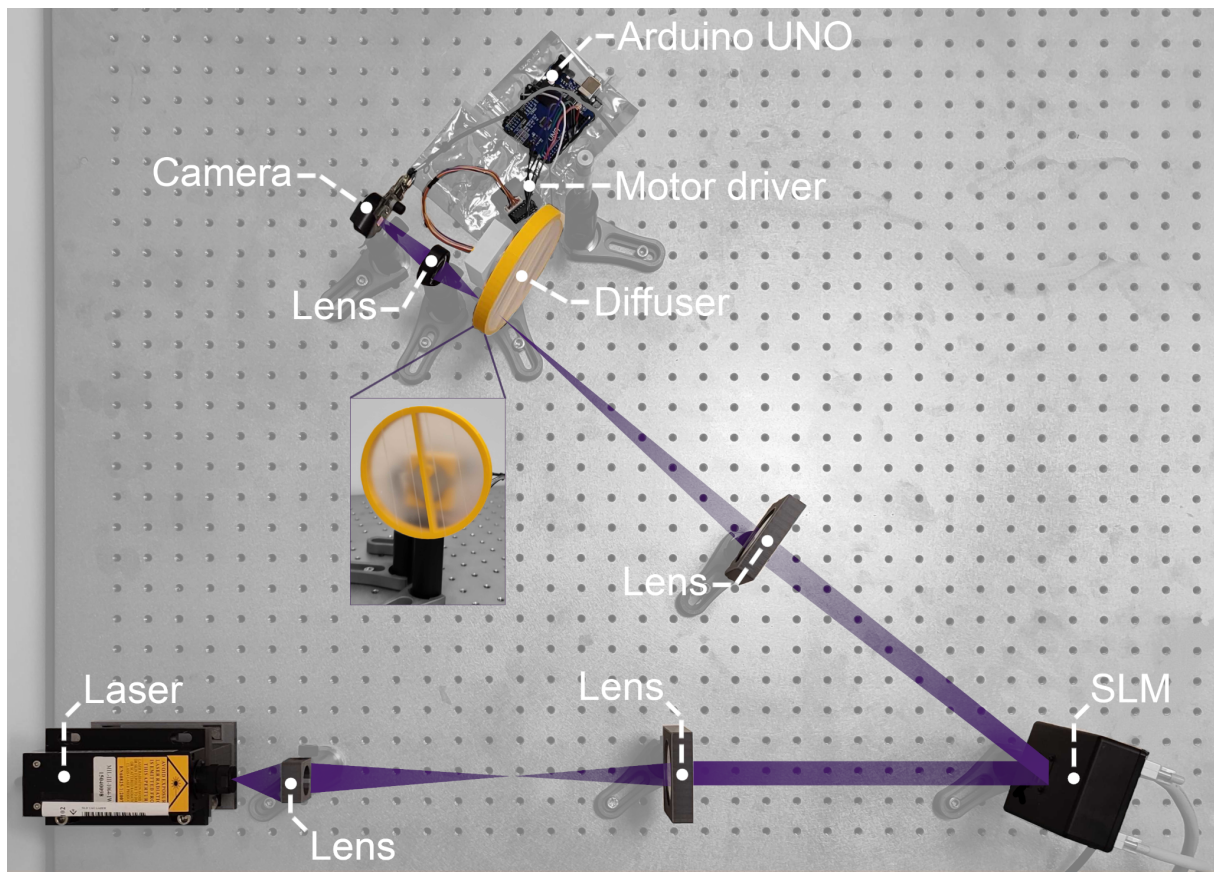


Figure S1: The experimental setup used for genetically programming random projection kernels. A disk covered with adhesive tape is used as the scattering medium, as seen on the inset. SLM: Spatial light modulator.

Supplementary Discussion 4: Hardware Flow Control and Software Flow Control

In Fig. S2 we illustrate our computing platform as a diagram including the interaction between the optical and electrical layers and the flow of our genetic algorithm-based scattering medium optimization.

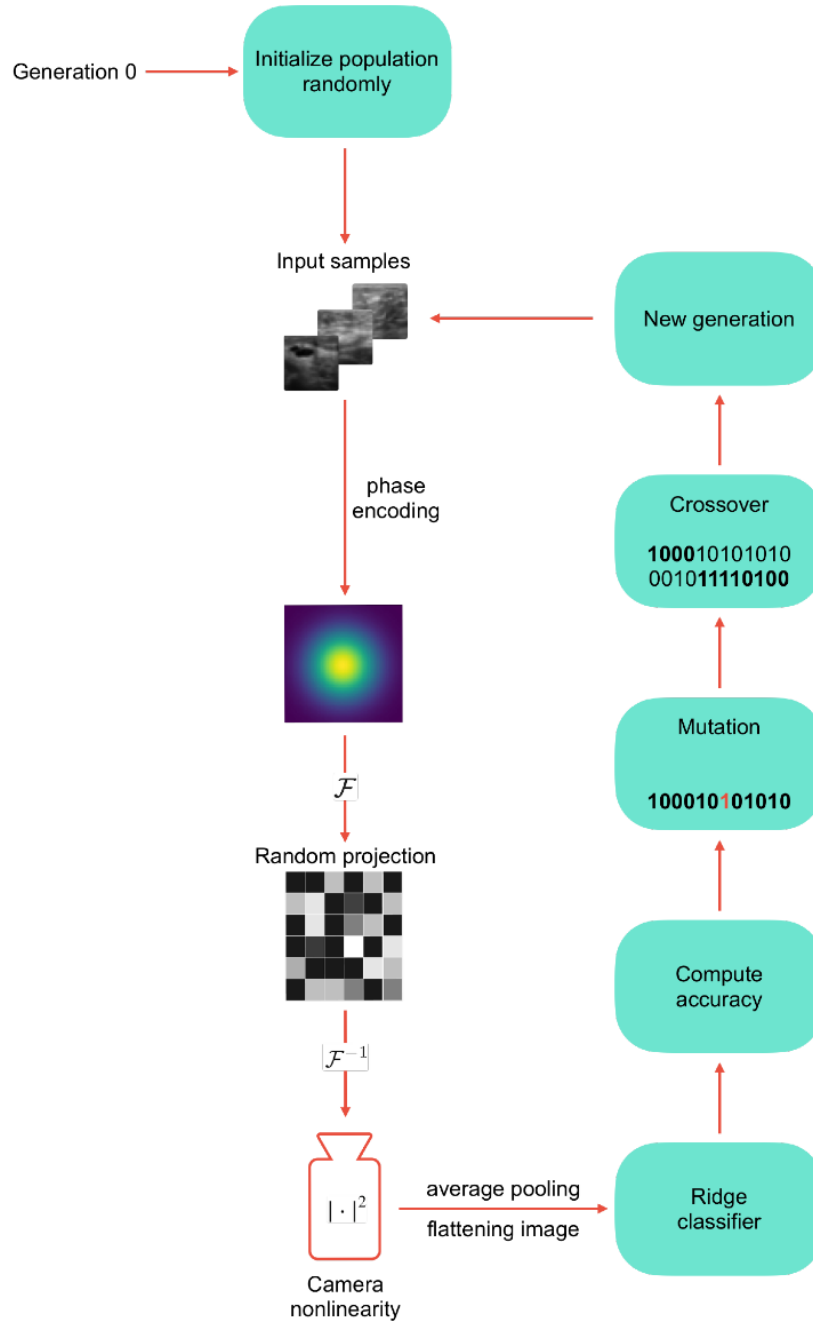


Figure S2: The hardware and software flow control of our proposal. The optical steps are illustrated with images and the software part is illustrated in light blue. Genetic algorithm operations such as mutation and crossover are shown schematically.

Supplementary Discussion 5: Similarity Between Adjoint Random Projection Kernels

To see the similarity between different random projection kernels, we performed two additional experiments. First, we fixed a reference point on our diffuser (disk with the adhesive tape) and rotated it within a range of 9 steps in either direction, which corresponds to an angular span of 1.58° , and recorded the resulting image at each angular position. We hypothesized that, by calculating the similarity between any image and the reference image, we could get a measure of the relation between corresponding random projection kernels. Because of its popularity in image processing and deep learning, we used the structural similarity index measure (SSIM). Fig. S3 shows the average SSIM for five trials. As can be seen in the figure, the general trend manifests itself as a negative correlation between angular displacement (in absolute terms) and similarity.

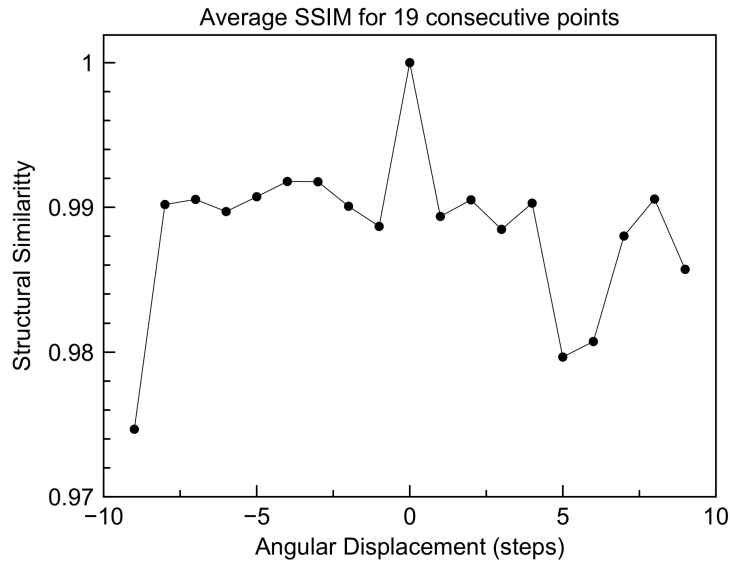


Figure S3: Average structural similarity index measure (SSIM) for consecutive points on the diffuser.

Second, we rotated the diffuser for 0° , 90° , 180° and 270° while a sample from the Fashion-MNIST dataset is displayed on the SLM. In Fig. S4 we show captured images and normalized average difference images together with SSIM values when the no rotation case is set as a reference.

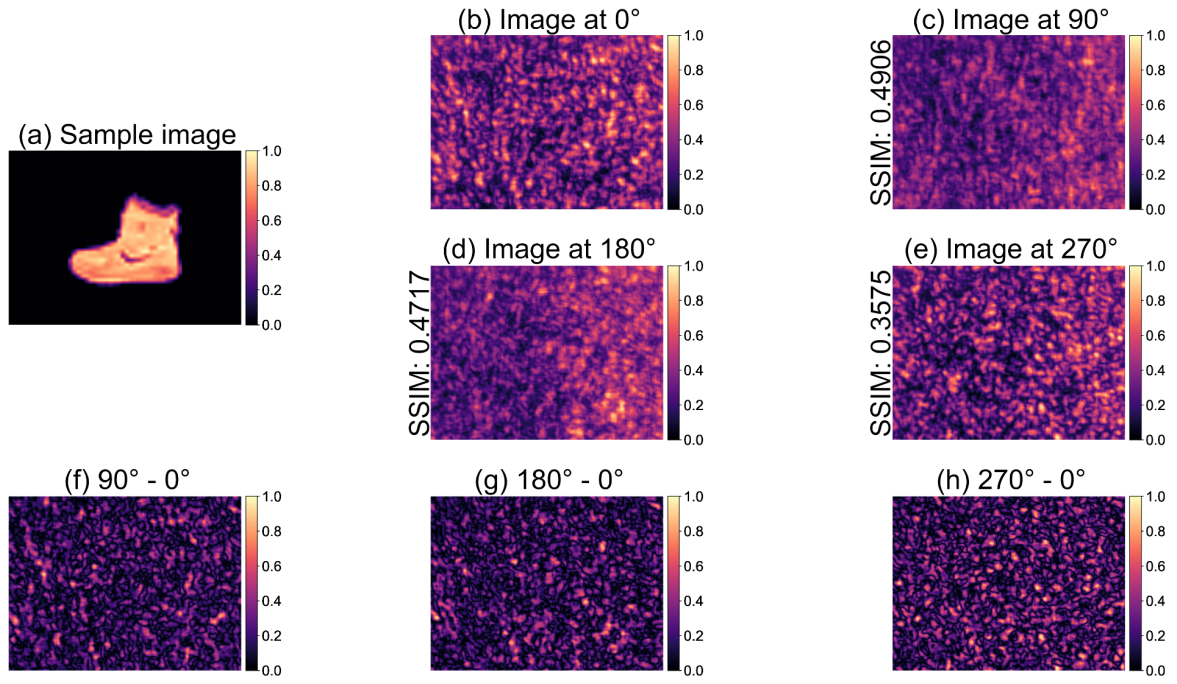


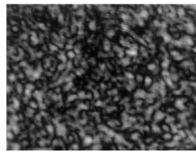
Figure S4: (a) Sample image from the Fashion MNIST dataset. (b-e) Captured images when the diffuser is rotated for 0°, 90°, 180° and 270°. Structural similarity index measure (SSIM) values are given to the left of the images when applicable. (f-h) Difference images when the no rotation case is considered as a reference.

Supplementary Discussion 6: Effects of Processing Input Images with Consecutive Random Projection Kernels

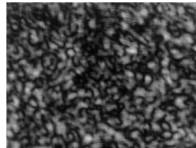
Fig. S5 shows captures corresponding to processing the same input image with 7 consecutive random projection kernels.

$\Delta\theta$ Capture

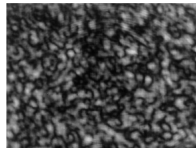
-3



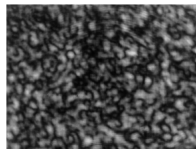
-2



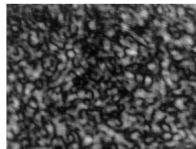
-1



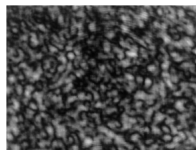
0



1



2



3

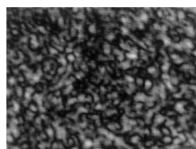


Figure S5: Captures corresponding to 7 consecutive random projection kernels. $\Delta\theta$ shows the difference in stepper motor units when the zero case is considered as a reference. Positive and negative $\Delta\theta$ values correspond to clockwise and counterclockwise rotation, respectively.

Supplementary Discussion 7: Further Results on the Subset Method

As discussed in the main text, accuracy results related to the COVID-19 X-Ray dataset were obtained using a subset method. We also performed GA on the full dataset to be able to assess the effectiveness of the subset method i.e., when all 2481 samples are randomly mapped. For the whole dataset, we obtained an accuracy of 91.15% compared to 90.14% obtained from 300 samples, which shows the approximation capability of the subset method. Fig. S6 shows the confusion matrices and the evolution of the accuracy in this setting.

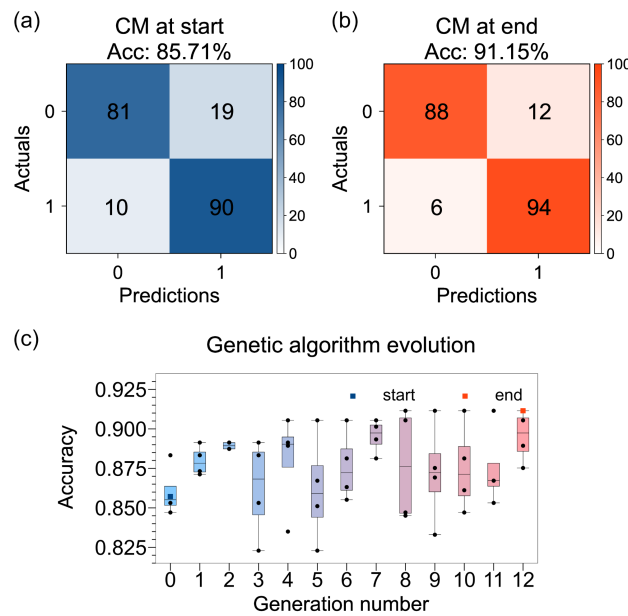


Figure S6: Results for the COVID-19 X-Ray dataset when the full set is used for genetic algorithm (GA). (a) Confusion matrix (CM) corresponding to the first GA iteration. (b) Confusion matrix obtained at the end of GA. (c) Evolution of the accuracy during GA.

Supplementary Discussion 8: Performance on the Fashion MNIST Dataset

We have also tested the popular Fashion MNIST dataset with 70,000 samples(2) to benchmark the performance of our programmable optical random projection scheme and to evaluate our method's performance for multilabel classification. Similar to the COVID-19 X-Ray dataset, we utilize generating a subset to decrease the programming time of our optical neural network. Thus, a subset of randomly selected 3000 samples with an 80/20 train-test split is created from 70,000 samples of the Fashion MNIST dataset. To set a baseline for our method, we apply the ridge classification over the selected subset and obtain 73.83% accuracy. At the beginning of the programming, after the first GA iteration, an accuracy of 71.33% is obtained. At the end of the programming, this classification accuracy is increased to 81.00% (Fig. S7b), resulting in an approximately 10% improvement. The entire Fashion MNIST dataset with 60,000

training and 10,000 test samples is optically processed for the programmed condition, and the classification accuracy of 83.06% is achieved. The row-wise normalized confusion matrices captured during the evolution of GA (Fig. S7a, c) and inference (Fig. S7d) further supported our claim that programming random neural networks effectively decreases classification errors.

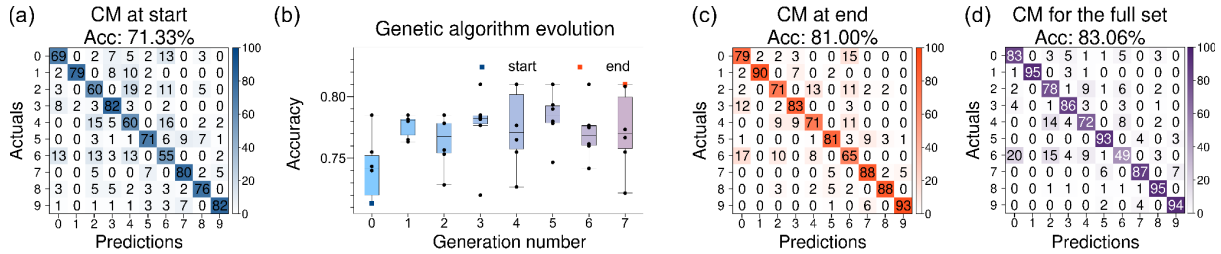


Figure S7: Experimental learning results for the Fashion MNIST dataset when a subset of the full dataset is used for genetic algorithm (GA). (a) Confusion matrix (CM) corresponding to the first GA iteration. (b) Evolution of the accuracy during GA. (c) CM obtained at the end of GA. (d) CM corresponds to the full dataset when all the samples are passed through the optimal angular position yielded by GA.

Supplementary Discussion 9: Performance on High Resolution Multilabel Classification Problems

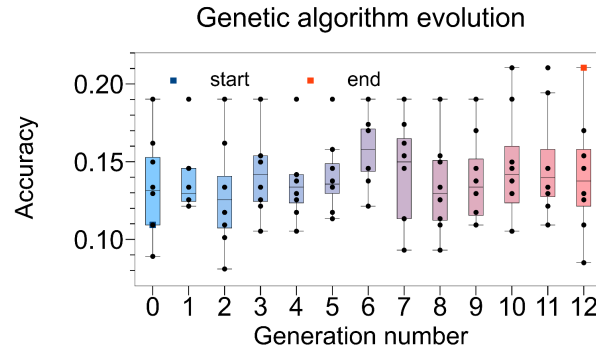
To better support our approach in terms of scalability in resolution, we perform additional multilabel classification experiments with the RS_C11 dataset (6) with 1232 samples each having 512 x 512 pixels resolution and the Aerial Image Dataset (AID) (7) having 10,000 samples each having 600 x 600 pixels resolution, both of which fit in our SLM. The task is to classify satellite images based on the scene categories they represent. There are 11 and 30 categories for RS_C11 and AID, respectively. An 80/20 train/test split is used for both datasets.

Fig. S8 shows the evolution of classification accuracy during GA iterations for both datasets, where confusion matrices are not visualized because of the high number of classes. Without any optical processing, the RS_C11 dataset yields 12.96% accuracy when ridge classification is solely used. The first GA iteration then yields 10.93%, which is further optimized to be 21.05%. The results for the RS_C11 dataset show the superiority of our approach compared to ridge classification.

For AID, we employed the same subsetting scheme that was discussed for the COVID-19 X-Ray and the Fashion MNIST datasets because of the high number of samples. In other words, we collected a subset consisting of 3000 samples (100 samples per scene category) and let GA run to optimize this subset. As previously, before any optimization was performed, the ridge classification accuracy of the subset was found out to be 25.83%. At the first GA iteration, the optically processed subset produced an accuracy of 22.33% which is worse than the unprocessed ridge

classification accuracy. However, as can be seen from Fig. S8b, we ended up with 34.83% accuracy. After completing GA iterations, we used the optimized random projection kernel to obtain 25.95% accuracy compared to the unprocessed accuracy of 12.9% for the whole dataset. Further comparisons for both datasets are given in Table S2.

(a) RS_C11 Dataset



(b) Aerial Image Dataset

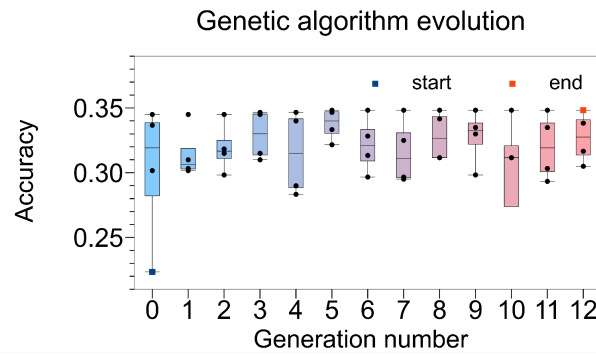


Figure S8: Evolution of classification accuracy for the high resolution multilabel geospatial (a) RS_C11 and (b) Aerial Image datasets.

Supplementary Discussion 10: Effect of GA Parameters on Classification Accuracy

In this section, we investigate the effects of the number of generations and population size on classification performance for some of the datasets evaluated in the main text. Towards this end, for each dataset, we perform multiple experiments where the number of generations is fixed at $n = 12$ and the population size is swept as $p = 4, 6$, and 8. Fig. S9 plots the accuracy versus population size for this setting.

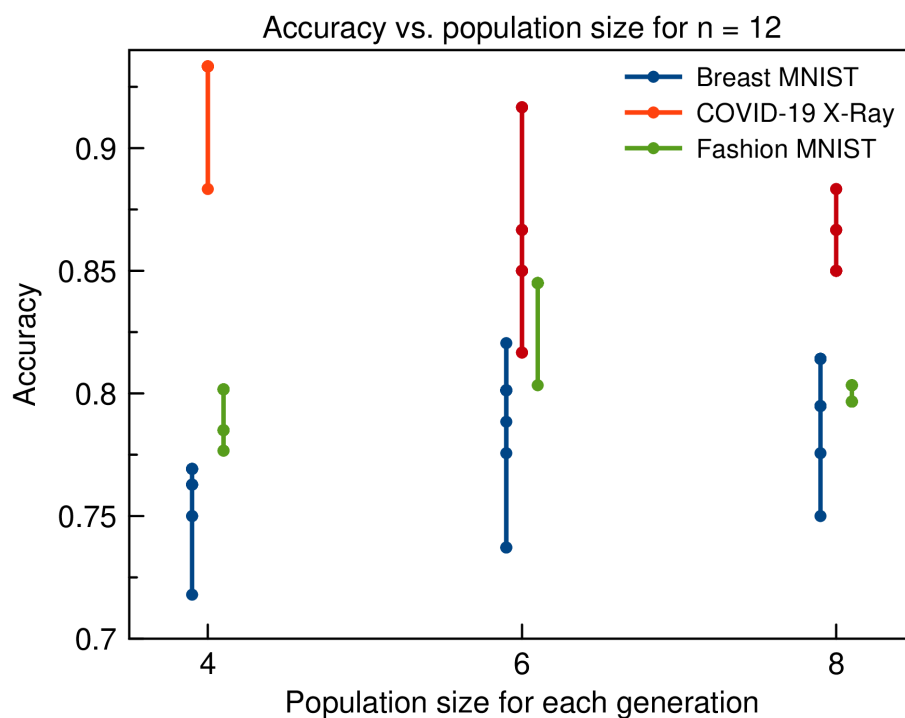


Figure S9: Accuracy values versus population size for a fixed number of generations of $n = 12$. Evolution of maximum accuracy through generations are given for population sizes of 4, 6, and 8.

Additionally, to examine how the accuracy changes under the condition that the number of random projection kernels evaluated during GA is kept constant at around 40, we performed experiments with $n = 5, p = 8$, and $n = 7, p = 6$. The evolution of maximum accuracy through generations is shown in Fig. S10 along with other experiments having different GA parameters. In the graph, while the line type (dashed or straight) differentiates between n values, different p values are color coded.

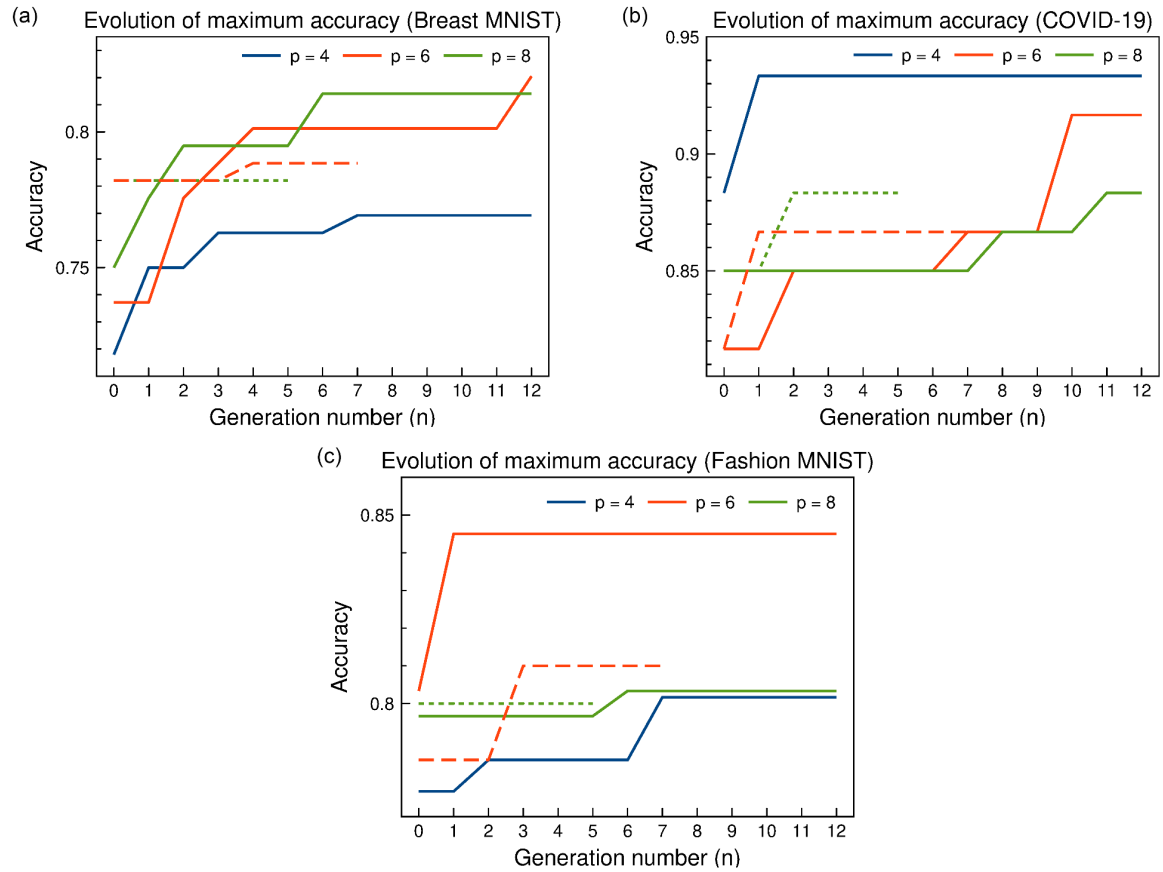


Figure S10: Evolution of maximum accuracy with different GA parameters for the (a) Breast MNIST, (b) COVID-19 X-Ray, and (c) Fashion MNIST datasets. p and n denote the population size and the number of generations, respectively.

Supplementary Discussion 11: Detailed Visualization of Clustering Performance

Fig. S11 shows 3D LDA results on the randomly-mapped subset of the Fashion MNIST dataset for the best GA iteration from various elevation and azimuthal angles.

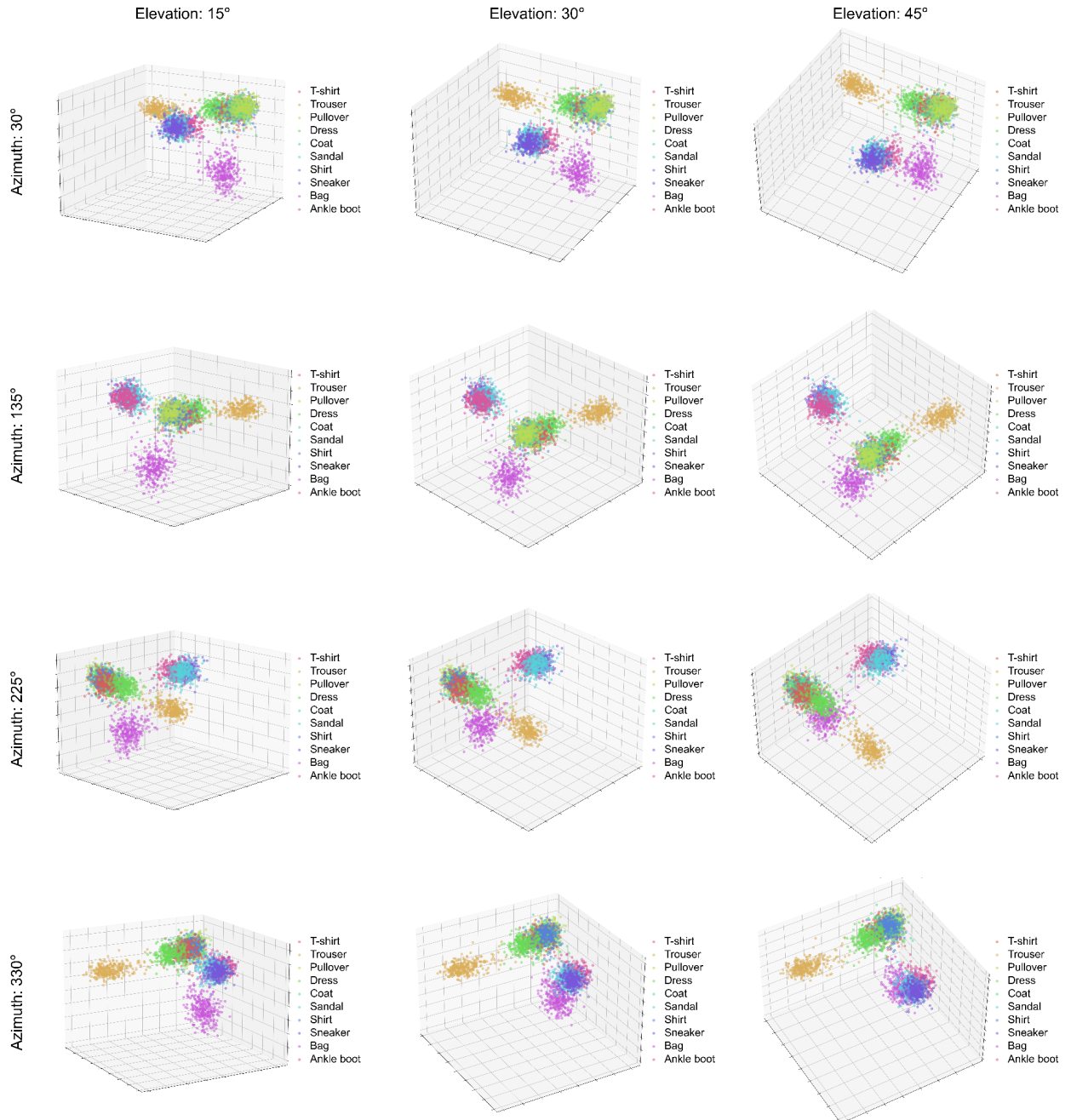


Figure S11: 3D linear discriminant analysis (LDA) results on the randomly-mapped Fashion MNIST dataset for the best genetic algorithm (GA) iteration for elevation angles of 15°, 30° and 45° and azimuthal angles of 30°, 135°, 225° and 330°.

Supplementary Discussion 12: Effects of Data Compression on Performance

When applying local averaging to randomly mapped data to reduce the number of pixels used for ridge classification, we fixed a pool size of (20,16). Fig. S12 shows alternative pool sizes and the corresponding number of pixels along with ridge classification accuracies.

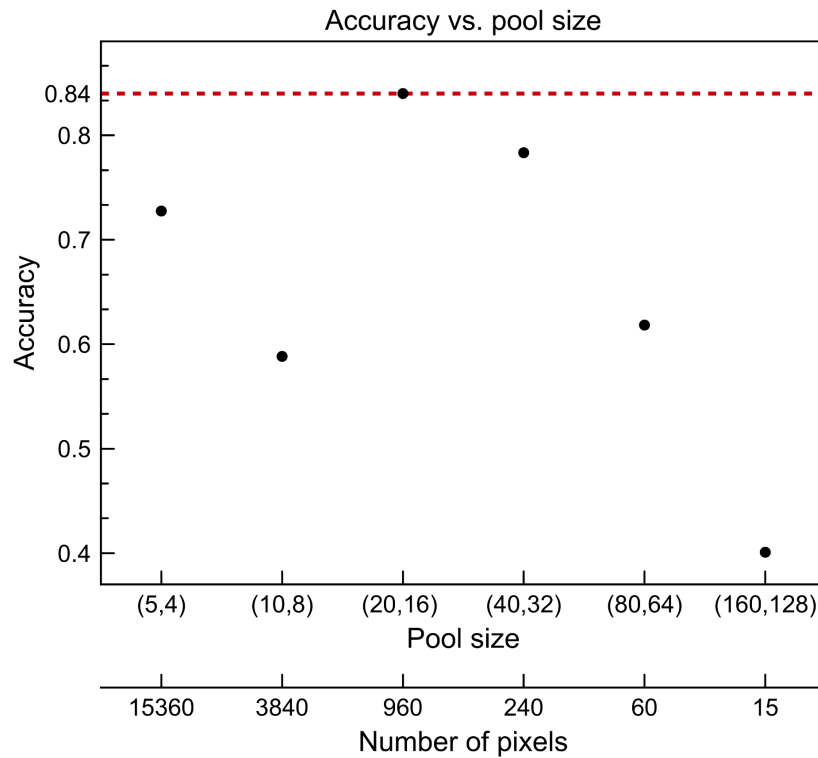


Figure S12: Ridge classification accuracies corresponding to different pool sizes used in local averaging for a randomly mapped subset of the Fashion MNIST dataset consisting of 6000 samples. The maximum accuracy is obtained for the aforementioned pool size of (20,16).

Data compression can also be carried out through reducing the bit depth in captures obtained from randomly mapped inputs. In our experimental setup, we used a camera capable of recording 8-bit grayscale images. Although practically not possible because of the way digital computers process data, we emulated a camera capable of representing images using less bits to test our approach in data-scarce settings by mapping the range 0 – 255 to a smaller range. If such an operation were possible, faster data rates would be enabled without sacrificing the classification accuracy. Fig. S13 shows GA results for different bit depths. We conclude that with suitable hardware tailored for modern machine vision tasks, our method can offer improvements in classification accuracy.

Although the positive effects of reducing the bit depth on the accuracy seem counterintuitive, we associated different levels of quantizing the resulting images with

nonlinearities of different power. As is known from neural networks, linear operations alone are not sufficient for generalization. Therefore, reducing the bit depth improves classification accuracy, and this reduction can be interpreted as employing ReLU (or sigmoid) functions of different slopes. However, further investigation is needed to uncover the exact relation between different bit depth levels and conventional nonlinearities.

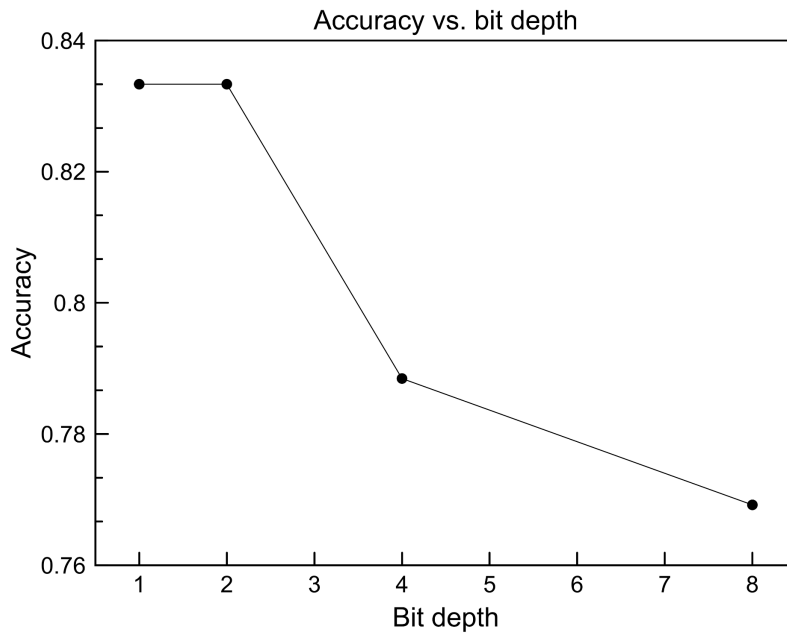


Figure S13: Dependence of ridge classification accuracy on the bit depth of captures in the Breast MNIST dataset. Separate experiments were carried out where 8-bit original captures were normalized and represented with 1, 2, and 4 bits. A maximum accuracy of 83.33% was obtained for bit depths of 1 and 2.

Table S1 demonstrates the effectiveness of our approach by comparing the classification accuracies obtained by (i) processing the randomly-mapped, 8-bit data and (ii) running GA from scratch (Fig. S13) to yield representations using less bits. These cases are denoted by the “accuracy without GA” and “accuracy with GA” rows, respectively. Without GA, after reducing 8-bit captures to 4-, 2-, and 1-bits and applying pooling and ridge classification, we observed small to no changes in classification accuracies, whereas when we let GA search for the optimal random mapping in different quantization schemes, we observed relatively higher improvements. This implies the adaptive nature of our approach to cases where high bitrates are desired when bit depth can be sacrificed. Modern machine vision tasks requiring real-time performance, therefore, could be a future application where we can test our programmable reservoir.

Table S1: Comparison of different quantization schemes in terms of classification accuracy with and without running genetic algorithm (GA) for the Breast MNIST dataset.

	8-bits	4-bits	2-bits	1-bit
Accuracy without GA	76.92%	76.92%	77.56%	76.28%
Accuracy with GA	-	78.85%	83.33%	83.33%

Supplementary Discussion 13: Benchmarking Accuracy Levels

To provide a broader perspective, we compared our approach with other popular approaches on the same datasets, taking into account the number of parameters used in each approach. This comparison is presented in Table S2 below. From the table, one can conclude that programmable optical random neural networks generalize well across different datasets when the number of parameters are constrained.

Table S2: Comparison of different approaches in terms of their classification accuracy and number of parameters for different datasets.

Dataset	Approach	Test accuracy	Number of parameters
Breast MNIST	Our approach	82.05%	961
	ResNet-18 (3)	86.3%	~11 million
	Ridge regression	66.67%	784
	PCA + Ridge	72.44%	2
COVID-19 X-Ray	Our approach	90.14%	961
	ResNet-101 (4)	95.58%	~44.5 million
	Ridge regression	74.85%	1000
	PCA + Ridge	60.97%	2
Fashion MNIST	Our approach	83.06%	961
	ResNet-18 (5)	94.9%	~11 million
	Ridge regression	81.13%	784

	PCA + Ridge	39.68%	2
Retinal disease classification	Our approach	87.70%	961
	Vanilla CNN	Not trainable	~468.9 million
	Ridge regression	85.65%	960
	PCA + Ridge	91.73%	2
RS_C11	Our approach	21.05%	961
	Ridge regression	8.91%	1024
	PCA + Ridge	20.65%	2
AID	Our approach	25.95%	961
	VGG-16 (7)	~90%	~138 million
	Ridge regression	12.9%	900
	PCA + Ridge	7.45%	2

Supplementary Discussion 14: Energy Consumption Across Computing Platforms

We now provide quantitative metrics regarding the energy consumption in our optical computing platform and conventional GPU-based computing platforms. In our setup, energy consumption is only due to the light source, spatial light modulator, stepper motor, camera, and the CPU used to implement ridge classification, which have power ratings of 4.6 W, 50 W, 1.2 W, 1.1 W, and 25 W, respectively, totalling 81.9 W. Meanwhile, an NVIDIA RTX 4070 graphics card draws 200 W of power.

References

1. GitHub - ugurtegin/MMF_RNN_Reuse: Reusability report: Predicting spatiotemporal nonlinear dynamics in multimode fibre optics with a recurrent neural network — github.com [Internet]. Available from: https://github.com/ugurtegin/MMF_RNN_Reuse
2. Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. 2017.
3. Yang J, Shi R, Wei D, Liu Z, Zhao L, Ke B, et al. MedMNIST v2-A large-scale lightweight

- benchmark for 2D and 3D biomedical image classification. *Sci Data*. 2023;10(1):41.
4. Covid-19 Binary Classification | ResNet101 | 96% [Internet]. Available from: <https://www.kaggle.com/code/ahmedtronic/covid-19-binary-classification-resnet101-96>
 5. GitHub - zalando research/fashion-mnist: A MNIST-like fashion product database. Benchmark — github.com [Internet]. Available from: <https://github.com/zalando research/fashion-mnist>
 6. Zhao L, Tang P, Huo L. Feature significance-based multibag-of-visual-words model for remote sensing image scene classification. *J Appl Remote Sens*. 2016;10(3):035004–035004.
 7. Xia GS, Hu J, Hu F, Shi B, Bai X, Zhong Y, et al. AID: A benchmark data set for performance evaluation of aerial scene classification. *IEEE Trans Geosci Remote Sens*. 2017;55(7):3965–81.