OPTICS

Transferable learning on analog hardware

Sri Krishna Vadlamani[1]*, Dirk Englund[1], Ryan Hamerly[1,2]

While analog neural network (NN) accelerators promise massive energy and time savings, an important challenge is to make them robust to static fabrication error. Present-day training methods for programmable photonic interferometer circuits, a leading analog NN platform, do not produce networks that perform well in the presence of static hardware errors. Moreover, existing hardware error correction techniques either require individual retraining of every analog NN (which is impractical in an edge setting with millions of devices), place stringent demands on component quality, or introduce hardware overhead. We solve all three problems by introducing one-time error-aware training techniques that produce robust NNs that match the performance of ideal hardware and can be exactly transferred to arbitrary highly faulty photonic NNs with hardware errors up to five times larger than present-day fabrication tolerances.

INTRODUCTION

Intense research over the past decade has demonstrated that neural networks (NNs) have a remarkable capacity to learn patterns and provide state-of-the-art performance in an astounding variety of artificial intelligence (AI) tasks (*1*–*4*). Artificial NNs, such as feedforward, recurrent, and residual networks, are parameterized functions that map input vectors to output vectors by performing successive matrix multiplications and elementwise nonlinear operations. The entries of the matrices, commonly called weights, are tuned to fit the function model to the training data for the given task. Top-end NNs today are composed of billions of weights and require massive amounts of data for training. The time and energy costs of training and inference on models of this scale have become a major challenge and have triggered a surge of interest in hardware AI accelerators (*5*, *6*), both digital and analog.

Analog accelerators promise tremendous energy and time savings (*6*, *7*), but one still needs to answer the universal criticism of analog circuits—that they can be unreliable as general-purpose computers because of both static hardware errors caused by manufacturing variations and inherent noise in the signals being processed. These problems persist in the particular case of analog optical NNs (ONNs). For instance, the splitting ratio of a typical fabricated beamsplitter deviates by 1 to 2% from 50-50 (*8*–*19*), which is sufficient to severely degrade the test accuracy of ONNs (*20*) composed of interconnected Mach-Zehnder interferometers (MZIs) (Fig. 1C). Hardware error correction techniques (*19*, *21*–*33*) applied to the hardware parameters provide substantial performance improvements but either require individual training/retraining of every ONN (Fig. 1D) (*21*–*28*), which is impractical in an edge setting with millions of devices, place stringent demands on component quality (*31*), or introduce hardware overhead (*19*, *32*, *33*). This is in sharp contrast to standard digital NNs (Fig. 1A), where training is performed only once and the resultant model can be deployed to any number of devices with no modification (Fig. 1B).

Here, we present a one-time error-aware software training technique that solves all three problems at once and brings analog NNs into the same league as digital NNs in terms of ease of model training and large-scale deployment. Our method outputs matrices that match the performance of trained ideal hardware and can be exactly transferred to any faulty ONN manufactured by a given process with no additional training or associated loss of performance (Fig. 1F). Moreover, the procedure does not add extra hardware to the existing ONN. We show through numerical simulations that the method tolerates hardware errors up to five times larger than present-day fabrication errors.

Our method is a combination of two important ideas: the error correction scheme of Bandyopadhyay *et al.* (*31*) and a form of engineering corner analysis (*34*). In more detail, it is known that splitter faults in an MZI shrink the set of unitary matrices that it can implement (Fig. 1E) (*32*); we introduce and train "maximally error-tolerant" MZI mesh–based ONNs that have the most faults and smallest expressivity for a given error level and show that the resultant matrices both have very high performance and can be exactly transferred to other ONNs with equal or smaller errors. In other words, our one-time training procedure allows us to train only one highly faulty ONN and freely transfer the resulting model to any number of edge ONNs with their own individual faults (Fig. 1F). Although we present results for feedforward MZI mesh–based ONNs, the training procedure is applicable to any MZI-based photonic circuit that permits implementation of the error correction scheme of (*31*). Our training procedure could, therefore, potentially find use in other applications of photonic circuits (*31*) such as quantum simulation (*14*, *35*–*38*), signal processing (*39*–*43*), and optimization (*44*).

Before we move on, it is important to clarify that the term "transferable learning" in the title of this paper is different from the "transfer learning" that is more common in the literature. Transfer learning is the paradigm in which models are first trained for a particular task and then fine-tuned to optimize performance on other distinct but related tasks. In the transferable learning of this paper, models are trained for a given task in such a way that they can be deployed onto any type of faulty hardware without any loss of performance on the same task—the same model for the same task is exactly "transferable" from one piece of faulty hardware to another. With this clarification in place, the rest of the paper is organized as follows: A summary of the optical hardware and the error correction scheme of Bandyopadhyay *et al.* (*31*) is given in the "ONN structure and error correction" section to make the paper

[1]Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. [2]NTT Research Inc., Sunnyvale, CA 94085, USA.
*Corresponding author. Email: srikv@mit.edu

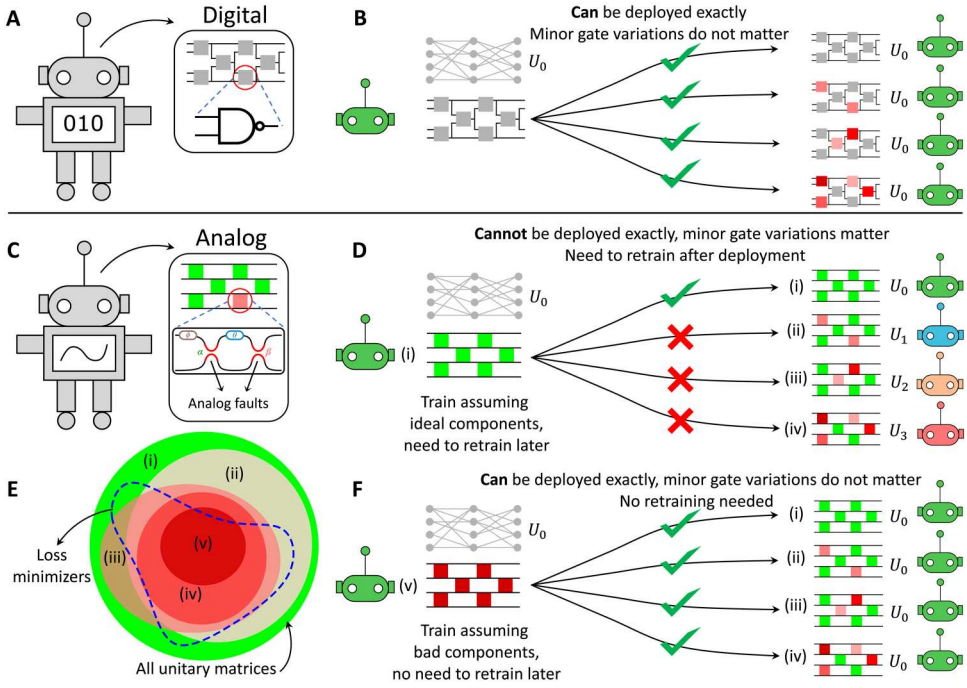

Fig. 1. Deploying trained models to digital and analog edge hardware. (**A**) Digital AI models run on digital logic circuits. (**B**) Training is performed once, and the resultant model U_0 is deployed directly to edge chips that implement U_0 faithfully irrespective of their individual gate characteristics. (**C**) Analog models run on imperfect analog hardware, optical unitary MZI meshes in this case. (**D**) Training on ideal hardware yields function U_0 that transfers exactly only to ideal chips. Transferring the raw parameters of U_0 to faulty chips leads to undesired functions U_1, U_2, and U_3 getting implemented. (**E**) Labels (i) to (v) refer to chips in panel F. Ideal meshes [chip (i)] can implement all unitary matrices; more faulty meshes implement fewer functions (green shrinks into red). However, there are good loss minimizers (dotted blue line) even within this restricted set. (**F**) Training on very faulty hardware [chip (v)] yields function U_0 that transfers exactly to all less faulty chips [(i) to (iv)] with no additional retraining.

self-contained; maximally error-tolerant MZI meshes are introduced in the "Maximally error-tolerant MZI meshes" section; one-time training and numerical results are presented in the "Transferable learning through one-time training" section; results and applications are presented in Discussion followed by Materials and Methods.

RESULTS

ONN structure and error correction

Any $N \times N$ unitary matrix can be decomposed (*45*, *46*) into a product of 2×2 unitary matrices and an $N \times N$ diagonal matrix D of complex phase shifts. The 2×2 unitaries are implemented in hardware by MZIs, while separate phase shifters implement the diagonal matrix (see the circuit between the two nonlinear blocks in Fig. 2A). Each MZI has two phase shifters, θ and ϕ. Individual MZIs are connected in a mesh topology that is consistent with the chosen $N \times N$ unitary decomposition method. ONNs are constructed by interleaving individual $N \times N$ meshes with elementwise nonlinear operations $\sigma(\cdot)$. The nonlinear function implemented by such a network (*31*) is derived in section S1. Figure 2A depicts an ONN layer composed of a 4×4 rectangular Clements (*46*) mesh of MZIs.

One way to use ONNs is to train a digital model of an ideal ONN with perfect 50-50 beamsplitters and to program the resultant optimal phase shifts θ and ϕ of all the MZIs and the diagonal matrices D into the hardware for inference. However, as mentioned previously, beamsplitter errors arising from process variation cause a mismatch between the digital model and the model implemented by the hardware, leading to severe degradation of ONN test-time performance when ideal trained phase shifts are programmed into the faulty hardware with no modification (*20*). Correction of the trained phase shifts to account for hardware errors is, therefore, essential.

Published error correction procedures include global methods that adjust individual MZI phase angles using circuit-wide optimization (*21*–*27*), local methods (*29*–*31*) that do so using only device-level information, and hardware augmentation methods that introduce additional beamsplitters ("3-MZIs," discussed in the "Transferable learning through one-time training" section) (*32*) or both beamsplitters and phase shifters (*19*, *33*) into the system. Global methods can improve performance but can be impractical in edge computing settings where the same model needs to be operated on a large number of edge devices. Local methods apply readily to edge settings because they involve quick local adjustments, but they do not correct over a large splitting error range. Hardware augmentation methods such as the 3-MZI approach (*32*) correct over a very large error range but incur chip area costs due to the extra hardware. We present a one-time global training method here that readily applies to edge settings, has a large splitting error correction range, and involves no additional hardware overhead. Because our approach uses concepts derived in the local error correction scheme of (*31*), we provide a brief overview of their method next (sections S1 to S4 contain a detailed derivation of this method).

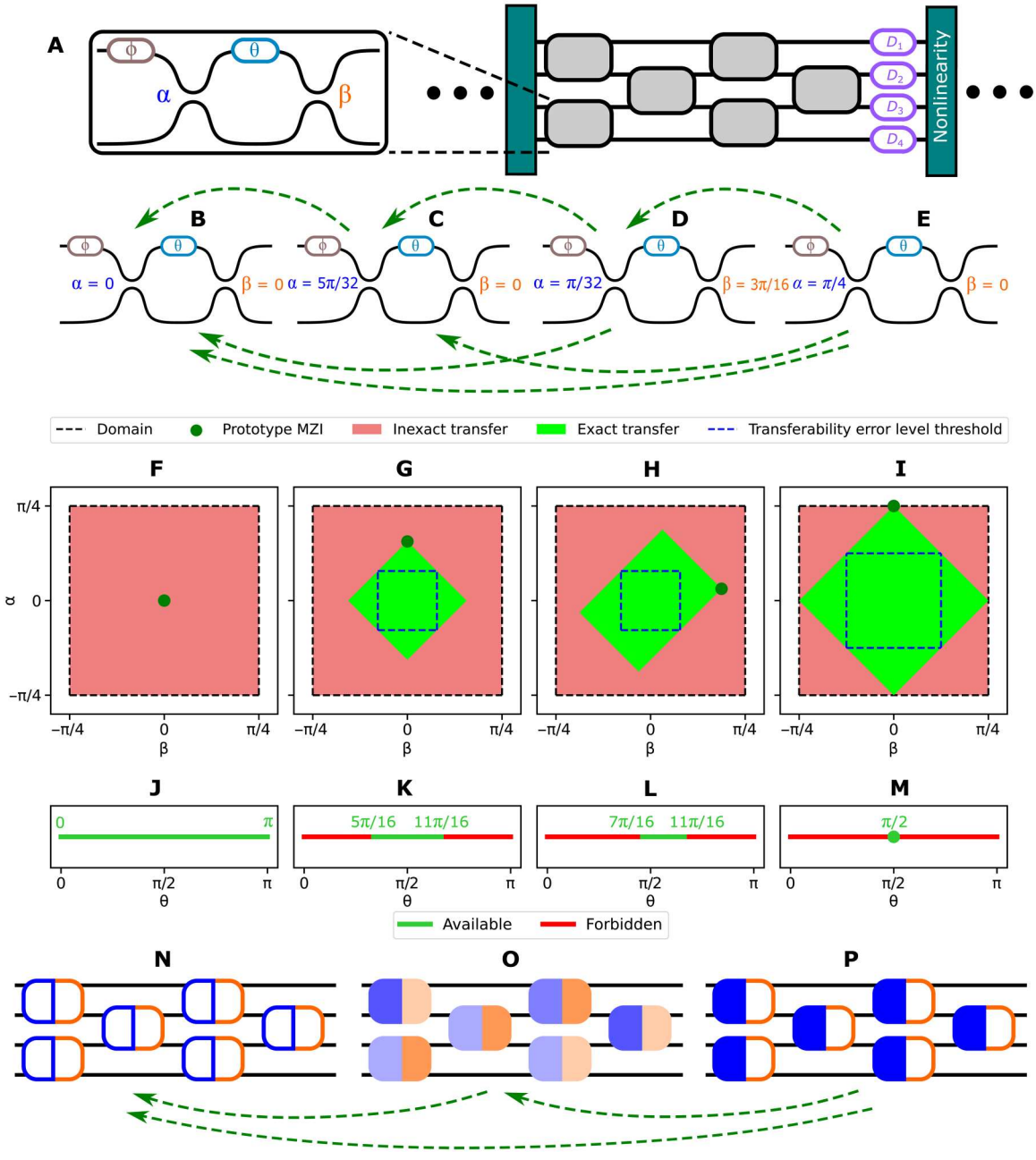

Fig. 2. More faulty meshes can be emulated exactly by less faulty ones. (**A**) A 4-by-4 Clements MZI ONN layer. (**B**) Ideal MZI and (**C** to **E**) faulty MZIs. Green arrows indicate that more faulty MZI functions can be transferred to less faulty MZIs. (**F** to **I**) Prototype MZI transfers only to MZIs inside the pale green rectangle. The blue dashed square is the largest error level $-\varepsilon \leq \alpha, \beta \leq \varepsilon$ up to which exact transfer occurs. Prototype MZI errors (α, β) of (F to I) are, in units of π, (0,0), (5/32,0), (1/32,3/16), and (1/4,0). (**J** to **M**) Prototype MZIs implement only a restricted range (green, "available") of ideal MZI matrix phase shifts θ. Representing MZI left beamsplitter error by blue fill, right by orange fill, (**N**) ideal mesh, (**O**) random faulty mesh, and (**P**) maximally error-tolerant mesh with $\alpha = 2\varepsilon$, $\beta = 0$.

The transfer function of an imperfect beamsplitter is

$$T^{\mathrm{bs}} = \begin{bmatrix} \cos(\pi/4+\alpha) & i\sin(\pi/4+\alpha) \\ i\sin(\pi/4+\alpha) & \cos(\pi/4+\alpha) \end{bmatrix} \quad (1)$$

where α is the "error angle" that captures the deviation from the ideal 50-50 ratio. Equation 1 reduces to the 50-50 case for $\alpha = 0$. Let the two error angles of a faulty MZI be denoted by α and β, respectively. Furthermore, let $T(\theta, \phi)$ and $T'(\theta, \phi, \alpha, \beta)$ represent the transfer functions of an ideal and a faulty MZI, respectively. Then, Bandyopadhyay *et al* (*31*) show that one can find an "error-corrected" set of phase shifts θ′, ϕ′, ψ_1, and ψ_2 such that

$$T(\theta, \phi) = \begin{pmatrix} e^{i\psi_1} & 0 \\ 0 & e^{i\psi_2} \end{pmatrix} \cdot T'(\theta', \phi', \alpha, \beta) \quad (2)$$

if and only if the ideal phase shift θ satisfies the following "error

correction condition"

$$2|\alpha+\beta| \leq \theta \leq \pi - 2|\alpha-\beta| \quad (3)$$

In words, Eq. 2 says that an ideal MZI with phase shifts θ and ϕ can be implemented by programming θ′ and ϕ′ into an imperfect MZI and adding phase shifts ψ_1 and ψ_2 to the two output arms if and only if the imperfect MZI satisfies the error correction condition (Eq. 3). The phase shifts θ′ and ϕ′ can be computed and programmed either via explicit mesh calibration or through self-configuration (*29*, *47*–*51*) (sections S3 to S5 contain further details on error correction and self-configuration). Equation 3 plays a crucial role in the definition of "maximally error-tolerant" MZI meshes, which is the subject of the next section.

Maximally error-tolerant MZI meshes

One approach to error-aware training is to calibrate the errors of the faulty hardware, construct a digital model of the system with the errors taken into account, train the digital model on the given task, and then port the resultant trained phase shifts back into the hardware. Alternatively, one could use the "physics-aware training" of (*52*), which eliminates the need for explicit error calibration by collecting a training set of input-output pairs of the hardware and training a digital NN ("digital twin") on it; the digital twin and physical hardware are together used to obtain a good model for the given task. The in situ training of (*27*, *28*, *53*), where backpropagation is performed in the hardware itself to obtain mesh-specific matrices, is yet another approach. All these approaches involve training each physical chip with its own individual errors separately (Fig. 1D), which is impractical in an edge computing setting with millions of edge devices. To solve this problem, we draw inspiration from corner analysis (*34*) and introduce the concept of maximally error-tolerant meshes—this idea enables us to train only one special mesh ("one-time training") for a given error level ε and transfer the resultant matrices over exactly to any other mesh (having the same geometry) with errors less than ε without any additional mesh-specific training or loss in performance.

More precisely, let us say that a fabrication process $\mathscr{P}$ is guaranteed to produce MZIs with errors $-\varepsilon \leq \alpha, \beta \leq \varepsilon$ for some error level $\varepsilon \geq 0$. An MZI (not produced by process $\mathscr{P}$) is maximally error-tolerant for error level ε if its errors satisfy $\alpha = 2\varepsilon$, $\beta = 0$. A mesh is maximally error-tolerant if all its MZIs are maximally error-tolerant.

Understanding maximally error-tolerant meshes

To understand the utility of maximally error-tolerant meshes, we return to the error correction condition (Eq. 3). The derivation of the condition implies that any faulty MZI with errors α, β can be exactly emulated by an ideal MZI with a θ that satisfies Eq. 3. The transfer function of this ideal MZI can, in turn, be exactly implemented by any other faulty MZI whose errors α′, β′ satisfy

$$|\alpha'+\beta'| \leq |\alpha+\beta| \text{ and } |\alpha'-\beta'| \leq |\alpha-\beta| \quad (4)$$

This is because this condition, together with the true statement $2|\alpha+\beta| \leq \theta \leq \pi - 2|\alpha-\beta|$, automatically implies $2|\alpha'+\beta'| \leq \theta \leq \pi - 2|\alpha'-\beta'|$. A corollary of this result is that a trained maximally error-tolerant MZI at error level ε can be exactly emulated by any faulty MZI whose errors α, β satisfy $|\alpha+\beta| \leq 2\varepsilon$ and $|\alpha-\beta| \leq 2\varepsilon$. This set includes all faulty MZIs with errors (α, β) that lie in the square bounded by the vertices (±ε, ±ε), that is, all the MZIs produced by the fabrication process $\mathscr{P}$ under consideration.

In the sequel, we shall refer to the MZI with errors α, β as a "prototype" for all other MZIs whose errors α′, β′ are smaller in the sense of Eq. 4. Figure 2 (B to E) depicts four example MZIs that will be treated as protoype MZIs in this discussion. Figure 2 (F to I) shows that prototype MZIs (dark green dot) can be emulated exactly by all MZIs in the α, β error phase space that satisfy Eq. 4 ("region of transferability," pale green rectangle) but not by MZIs that do not (red). The example prototype MZI errors are specifically chosen such that the prototype MZI of each panel ("panels" in this discussion refer to parts of Fig. 2) lies within the region of transferability of the prototype MZI of each panel to its right. Therefore, the transfer matrix of panel E can be exactly implemented by all the MZIs to its left; the transferability of matrices between meshes is indicated by green dashed arrows. Only the prototype MZIs of panels C and E are maximally error tolerant. Figure 2 (J to M) depicts the range of ideal MZI θ phase shifts that are implementable by the prototype MZIs in panels B to E; the more faulty an MZI is, the less expressive it is.

The blue dashed squares [with corners (±ε, ±ε)] inside the green rectangles of Fig. 2 (F to I) mark the largest error level ε ("transferability error level threshold") up to which the prototype MZI of that panel is transferable. Panels G and H have blue dashed squares of the same size. However, the maximally error-tolerant MZI of panel C explores a wider range of ideal θ phase shifts (panel K) than MZI D (phase shifts in panel L). Therefore, it is clear that, for any given error level ε, maximally error-tolerant MZIs apply less restrictions on the search space θ than any other prototype MZI.

The discussion above immediately suggests a one-time training procedure: train a maximally error-tolerant MZI mesh only once at a high enough error level ε, and one can then readily transfer the trained model exactly to any other MZI mesh that has errors smaller than ε. How the maximally error-tolerant meshes are trained is the subject of the next subsection. Once the training is done, the transfer of the trained phase shifts from the maximally error-tolerant mesh to a less faulty one may be performed in two steps: (i) translate the phase shifts of the more faulty mesh to an ideal mesh using the "inverse" of the error correction of (*31*) (see section S4) and (ii) translate the ideal phase shifts to the less faulty mesh using "vanilla" error correction (*31*). Both steps are guaranteed to work exactly. Alternatively, one could use the self-configuration of (*47*, *48*) to directly program the matrix of the more faulty mesh into the less faulty one in a single step. The all-important role played by the error correction condition (Eq. 3) in the above discussion implies that maximally error-tolerant meshes can only be constructed if the underlying mesh geometry permits implementation of the error correction scheme of (*31*). This includes all types of feedforward MZI mesh networks.

Transferable learning through one-time training

One-time training simply consists of training maximally error-tolerant meshes for a given error level; the resulting matrices can then be transferred directly to any other arbitrary mesh at a lower error level with no additional retraining (hence the term "one-time"). We present two approaches to transferable learning of maximally error-tolerant meshes: (i) a direct training approach where maximally error-tolerant meshes are trained separately for each given error level and (ii) a "transfer training" approach where the trained raw phase shifts of a maximally error-tolerant mesh at one error level are

used as the starting point for training a maximally error-tolerant mesh at the next higher error level. Regardless of which method is chosen, the model obtained upon the completion of training can be freely deployed to any faulty network at a lower error level with no additional retraining. Our simulations (Fig. 3B) used the neurophox (*30*) and meshes (*54*) packages and were performed on an NVIDIA Tesla K40 GPU and the Engaging computing cluster at the Massachusetts Institute of Technology (MIT). Results from two-layer Clements mesh-based ONNs (Fig. 3A) are presented in Fig. 3B for the Modified National Institute of Standards and Technology (MNIST) (*55*) (digit), FashionMNIST (*56*) (clothing), and KMNIST (*57*) (Japanese character) classification tasks. The raw images of all datasets are low-pass–filtered (Fig. 3A); the 256 and 400 slowest spatial frequencies (labeled "inputsize" in Fig. 3B) are retained to enable detection of input size dependence of one-time training. Because the Fourier transform operation can be cast as a unitary operation on the one-dimensional unrolled image, the low-pass preprocessing can be done entirely optically on-chip through another MZI mesh.

Figure 3B depicts three baselines: (i) the uncorrected case (red), where error-free meshes were trained and the resultant phase shifts were directly programmed, with no error correction, into faulty meshes; (ii) the corrected case (green), where the ideal trained phase shifts were first error-corrected according to (*31*) and then fed into faulty meshes; and (iii) the 3-MZI case (orange). 3-MZIs are standard MZIs with an additional beamsplitter (*32*). The ideal trained matrices are fed into faulty 3-MZI meshes via self-configuration (*47*). The bold lines in Fig. 3B are the medians over independent runs, while the paler sheath around the bold line represents the interquartile range (IQR). These baselines are compared against two varieties of transferable learning: one-time direct training and one-time transfer training.

Transferable learning—Direct training of maximally error-tolerant meshes

In this approach, the phase shifts of maximally error-tolerant meshes are trained from randomly initialized starting points θ and ϕ for each error percentage point between 0 and 35% error level; the results are plotted in Fig. 3B in blue. In both the MNIST and FashionMNIST tasks, maximally error-tolerant mesh training matches or exceeds the performance of error correction (green) and the 3-MZI mesh (orange) up to 35% error level for both mesh sizes considered. There is a curious improvement in the performance that direct training achieves compared to the 3-MZI mesh on the FashionMNIST task that one could try to attribute to a regularization caused by the fact that faulty meshes implement fewer unitaries than ideal meshes. That this is not a general phenomenon is immediately borne out by the substantially poorer test accuracy of direct training on KMNIST although it is still within 1% of the 3-MZI performance up to 10% error level.

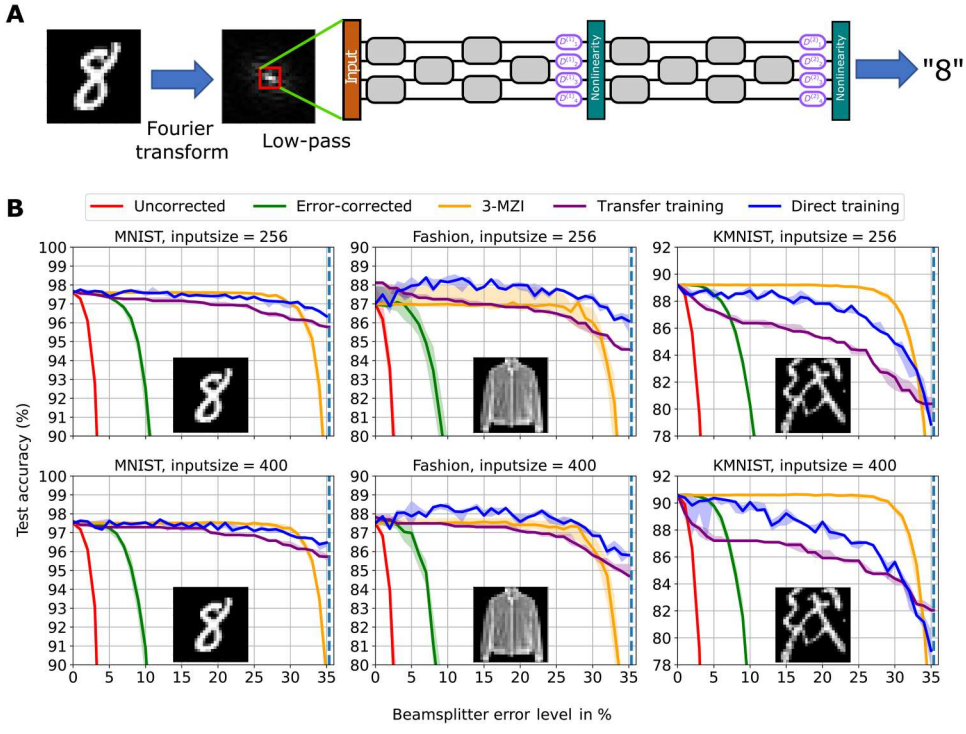

Fig. 3. Network structure and results. (**A**) Input images are low-pass–filtered and passed into a two-layer mesh-based ONN (size 4 illustrated for convenience). (**B**) Performance (median, 25th and 75th percentiles) of one-time training on the MNIST, FashionMNIST, and KMNIST datasets, for input sizes of 256 and 400. Insets show example images only; the accuracies are computed over the full test set. Five maximally error-tolerant models were trained for each error level from 0 to 35% (step of 1%); the test accuracies are plotted in blue (one-time direct training). The test accuracies obtained from direct transfer of ideal model weights to random faulty standard MZI networks are plotted in red, results of transfer after error correction are in green, and results of error-corrected transfer to faulty 3-MZI networks are in yellow. Results of repeatedly transferring lower error level network phase shifts to a higher error level network and retraining for two epochs each time are in purple (one-time transfer training).

Transferable learning—Transfer training of maximally error-tolerant meshes

In this approach, instead of training a maximally error-tolerant mesh for p% error level from randomly initialized phase shifts as in direct training, we use the raw, uncorrected phase shifts of a trained maximally error-tolerant mesh at $(p - 1)$% error level as the starting point. Because model training does not begin from a random starting point, fewer epochs are needed to get to a good set of phase shifts at each higher error level. The model obtained upon completion of transfer training (say, upon reaching $p = 35$) can then be freely deployed onto arbitrarily faulty hardware at a lower error level with no additional retraining. The accuracy curves for transfer training are plotted in purple in Fig. 3B. More information about curve smoothing is provided in Materials and Methods.

The results indicate that transfer training is nearly as good as direct training and the 3-MZI mesh on both the MNIST and FashionMNIST tasks for both input sizes. On the other hand, the results for KMNIST, which is known to be a difficult dataset (*57*), are worse than even the error-corrected green curve. Transfer training was rerun for KMNIST with an increased number of epochs of training for every increase in error level; the substantially improved performance, which now matches direct training, is depicted in pink in Fig. 4A.

Unbalanced MZI losses

While the beamsplitter splitting errors considered so far preserve the unitarity of the mesh, unbalanced losses in the MZI arms can render the mesh transfer function nonunitary. We worked with KMNIST with input size of 256 to demonstrate that unbalanced losses have negligible influence on the test performance of ONNs.

Figure 4B reports the evolution of test accuracies as random unbalanced MZI losses are progressively introduced into (i) a trained network composed of perfect 50-50 beamsplitters (green), (ii) a trained maximally error-tolerant network at 10% beamsplitter error level (blue), and (iii) a trained network with random beamsplitter errors at 10% error level (red). Because the networks in all three cases were not retrained to adapt to the introduced loss, the results in Fig. 4B demonstrate that models trained on lossless meshes are robust in the presence of unbalanced losses in the actual hardware. The typical loss values for the phase shifters, beamsplitters, and component sizes were taken from (*31*, *47*, *58*); further details are provided in the "Unbalanced MZI loss data" section.

DISCUSSION

ONNs are a leading analog accelerator platform for large-scale machine learning. However, their performance degrades markedly in the presence of static MZI beamsplitter errors (*20*). Existing error correction procedures are either impractical in large-scale edge settings, applicable over small beamsplitter error ranges, or involve additional hardware overhead. Here, we presented a one-time error-aware training technique for MZI-based ONNs that tackles all these problems. The method matches ideal-hardware performance even in the presence of large static hardware phase errors up to five times larger than present-day fabrication tolerance. Moreover, it is transferable and one time, that is, the training is performed only once and the resultant matrices can be programmed directly into any number of arbitrary highly faulty photonic NNs in an edge setting with no additional retraining. Furthermore, the method uses only standard MZIs and does not require additional hardware.

Our key contribution was the introduction of a principled combination of two important ideas: error correction and engineering corner analysis. More specifically, we introduced the concept of a "maximally error-tolerant network," one in which every MZI has errors $\alpha = 2\varepsilon$, $\beta = 0$ for some $\varepsilon > 0$, and showed that matrices obtained by training such a network yield excellent test performance over a very large range of ε. Furthermore, the trained matrices can be exactly ported, using self-configuration or error correction, onto other MZI networks (with the same underlying geometry) whose splitting error angles α, β all lie in the range $[-\varepsilon, \varepsilon]$ with no additional training and no loss of performance associated with the transfer.

We presented two variants of transferable learning: (i) "direct training" of a maximally error-tolerant network from a randomly

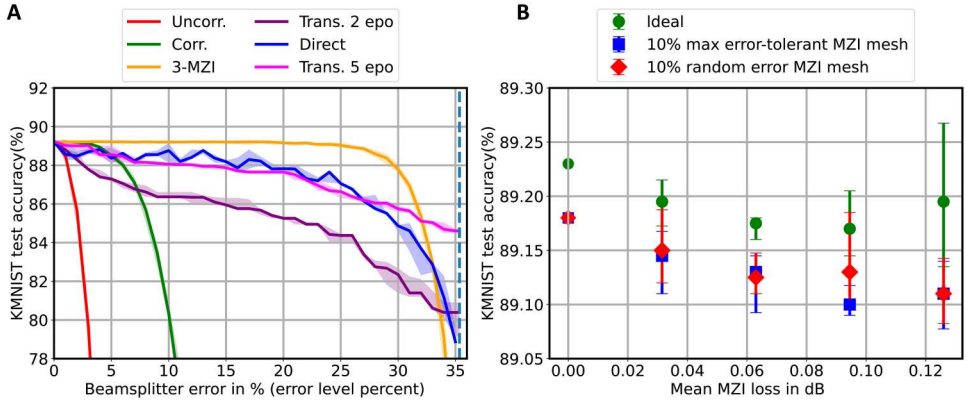

Fig. 4. Improved transfer training and robustness to unbalanced losses. (**A**) Using five epochs (pink) of training for every unit increase in error level allows transfer training to match direct training for KMNIST, input size of 256. (**B**) The addition of unbalanced MZI losses to trained ideal networks (green) or maximally error-tolerant networks at 10% error level (blue) or networks with random beamsplitter errors at 10% error level (red) leads to no degradation in test accuracy. For each type of mesh, and at each loss level, 10 lossy meshes were generated, and the medians and 25th and 75th percentiles of the test accuracies are plotted. Results are for KMNIST, input size of 256.

initialized starting point θ, ϕ for a given error level, and (ii) transfer training where one repeatedly transfers raw trained phase shifts of maximally error-tolerant networks at lower error levels to maximally error-tolerant networks at a slightly higher error level followed by a small amount of retraining. Numerical experimentation indicated that our method approached or achieved the large error tolerance of 3-MZI (*32*) networks on several benchmark tasks without incurring the additional hardware overhead of 3-MZIs. We also demonstrated that the typical unbalanced losses of interferometer chips have a negligible effect on the performance of our models, even at a high beamsplitter error level. While our study was based on feedforward MZI ONNs, the procedure can be applied to any MZI-based photonic circuit whose underlying geometry permits implementation of the error correction scheme of (*31*). More generally, we believe that the transferable learning method is applicable to any physical hardware, even nonoptical, that supports some type of error correction along with a kind of corner analysis that permits exact parameter transfer from a more faulty setup to a less faulty one.

MATERIALS AND METHODS

Hyperparameters and preprocessing

Each of the three datasets contains 70,000 monochrome images of size 28 × 28: 60,000 training images and 10,000 test images. Because it was previously observed that the higher spatial frequencies of MNIST images do not contain much information (*30*), the images were low-pass filtered by Fourier transforming them and selecting only a smaller square of Fourier components of side s centered at the origin. Incidentally, we observed that single-layer classifiers trained on the low-pass filtered Fourier components yielded higher test accuracy than those trained on the raw input images.

To probe the effectiveness of one-time training at different mesh sizes, we ran simulations for both $s = 16$ and $s = 20$, which correspond to 256 and 400 total input features (labeled inputsize in Fig. 3B), respectively. Our NNs were two-layered, each layer was a Clements unitary mesh, and the electro-optic nonlinearity of (*59*) was used between layers (Fig. 3A). The first 10 outputs of the output layer were treated as the label predictors. The standard cross-entropy loss and the Adam optimizer were used.

A closer inspection of Eq. 3 reveals a natural upper limit on the error levels that our method can tackle. A maximally error-tolerant MZI emulates ideal MZIs whose phase shift θ lies in the range $4\varepsilon \leq \theta \leq \pi - 4\varepsilon$. The lower and upper limits of this range coincide at $\varepsilon = \pi/8$ and the expressivity of a maximally error-tolerant MZI collapses to a single value of θ. For $\varepsilon > \pi/8$, there does not exist a single maximally error-tolerant MZI that transfers to all MZIs at that error level. Therefore, maximally error-tolerant MZIs are a meaningful concept only up to error level $\varepsilon = \pi/8$ (35.36%); all our results are plotted up to that error level only.

Baseline data

The performance of our transferable learning approach is compared against baseline data generated from the simulated transfer of the parameters of trained standard error-free mesh models onto faulty meshes with random beamsplitter error angles. While typical fabricated MZI-based circuits have random but spatially correlated beamsplitter error angles with a particular correlation distance, we use uncorrelated error angles in our simulations. This is because it was previously shown in (*48*) that spatial inter-MZI error angle correlations do not contribute to the deviation of the mesh from the target matrix if the target matrix is drawn from the Haar distribution.

To generate the data for the three baselines (uncorrected, corrected, and 3-MZI), five ideal error-free meshes with independent Haar-random initial phase-shift conditions were trained for 50 epochs each. Next, for each ideal model and error level ε (which corresponds to $100\frac{\sin(2\varepsilon)}{2}$ in percent, the quantity plotted on the x axis of Fig. 3B), five faulty meshes were generated with MZI error angles chosen independently and uniformly randomly from the range [−ε, ε]. The step size in the error level was 1%. The ideal matrices were then transferred to these faulty meshes, by the process indicated for each baseline in the "Transferable learning through one-time training" section, and the test accuracies were recorded. This yields five values at each error level for each ideal model. Because there are five ideal trained meshes, we have 25 test accuracies for each error level from 1 to 35%. The medians of these numbers are plotted as bold lines in Fig. 3B, while the IQR is represented as a paler sheath of the same color around the central bold line.

Maximally error-tolerant meshes—Direct training data

In this approach, the phase shifts of maximally error-tolerant meshes are trained from randomly initialized starting points θ, ϕ for each error level. For each percentage point between 0 and 35% error level, five maximally error-tolerant meshes were trained independently for 50 epochs, and the median and IQR of these five values are plotted in Fig. 3B in blue.

Maximally error-tolerant meshes—Transfer training data

In this approach, a maximally error-tolerant mesh at p% error level is trained using the raw, uncorrected phase shifts of a trained maximally error-tolerant mesh at $(p - 1)$% error level as the starting point. Because model training does not begin from a random starting point, fewer epochs are needed to get to a good set of phase shifts at the higher error level. In our implementation, we started out once again with the five ideal trained models that were previously used for the error correction and 3-MZI results. The uncorrected ideal model phase shifts are programmed into a maximally error-tolerant mesh at an error level of 1%, and this mesh is trained for two epochs. The resultant phase shifts are then fed directly into a maximally error-tolerant mesh at an error level of 2%, and two more epochs of training are performed. This training rate of two epochs for every percent increase in error level is maintained up to 35% error level, whereupon two more epochs of training are performed on a final mesh with 35.36% error level.

This procedure is performed with each of the five ideal trained models used as a starting point, yielding five models at each error level. The test accuracies of these models tend to be nonmonotonic, jagged functions of the error level, similar to the jagged blue curves of the direct trained models in Fig. 3B. The fact that the higher error level meshes can be emulated exactly by lower error level meshes suggests that one can make jagged accuracy curves monotonic by assigning to each error level the performance of the best model at the same or higher error level. This "curve smoothing" is computationally prohibitive for direct training because 37 × 50 = 1850 epochs are required to generate trained models for all error levels from 0 to 35.36%. Because it is likely that direct training will be applied to only a few error levels in a real-world setting, Fig. 3B

does not depict smoothed-out direct training results. On the other hand, because generating transfer trained models for the same error level range requires only 50 + 36 × 2 = 122 epochs, we smooth out the accuracy curves for transfer training and plot the median and IQR in purple in Fig. 3B.

For the improved five-epoch–per–step transfer training rerun on KMNIST, the number of epochs required (50 + 36 × 5 = 230) is still smaller than the cost of direct training for all error levels (1850 epochs).

Unbalanced MZI loss data

Figure 4B, which illustrates the effect of unbalanced MZI losses on network test accuracy, was generated using the component size and loss values reported in (*31*, *53*, *58*). Wilmart *et al.* (*58*) report an average Silicon-On-Insulator (SOI) waveguide loss of 2.1 ± 0.25 dB/cm when a "typical" fabrication recipe is used, and an average loss of 0.1 ± 0.04 dB/cm when a "state-of-the-art" recipe with an added H_2 thermal annealing step is used. While Bandyopadhyay *et al.* (*31*) mentions that the beamsplitters and titanium nitride–based thermal phase shifters are typically 100 and 400 μm long, respectively, the experimental demo of (*53*) uses 200-μm phase shifters. Figure 4B presents results for meshes with 200-μm-long thermal phase shifters, 100-μm-long beamsplitters, and mean waveguide losses of 0, 0.525, 1.05, 1.575, and 2.1 dB/cm [i.e., 0, 25, 50, 75, and 100% of the mean 2.1 dB/cm loss observed in (*58*); the loss variance for each case was obtained by similarly scaling the reported variance in (*58*)]. Per-MZI loss is assumed to follow a Gaussian distribution with the mean loss (on the *x* axis of Fig. 4B) and the per-MZI loss variance (not shown) being calculated from the loss values and the component lengths. Ten random lossy networks with no beamsplitter errors were generated at each mean loss level (with the loss of each MZI being sampled independently from the Gaussian), and the raw phase shifts of the two-layer lossless perfect MZI networks (with 50-50 beamsplitters) that were trained in earlier sections were programmed directly into the lossy networks; the resultant test accuracies are reported in green in Fig. 4B. The results of programming the raw phase shifts of trained lossless 10% maximally error-tolerant MZI networks into lossy 10% maximally error-tolerant MZI networks are shown in blue, while the results of self-configuration of the matrices of trained lossless 10% maximally error-tolerant MZI networks into lossy 10% randomly faulty MZI networks are shown in red.

Supplementary Materials

This PDF file includes:

Section S1 to S5
Fig. S1

REFERENCES AND NOTES

1. D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, D. Hassabis, A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* **362**, 1140–1144 (2018).
2. T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language models are few-shot learners. *Advances in Neural Information Processing Systems*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, H. Lin, Eds. (Curran Associates Inc., 2020), pp. 1877–1901.
3. A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, M. Chen, Hierarchical text-conditional image generation with CLIP Latents. arXiv:2204.06125 (2022). https://doi.org/10.48550/arXiv.2204.06125.
4. J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, D. Hassabis, Highly accurate protein structure prediction with AlphaFold. *Nature* **596**, 583–589 (2021).
5. A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, J. Kepner, AI accelerator survey and trends. *2021 IEEE High Performance Extreme Computing Conference (HPEC)* (IEEE, 2021), pp. 1–9.
6. Y. Shen, N. C. Harris, S. Skirlo, M. Prabhu, T. Baehr-Jones, M. Hochberg, X. Sun, S. Zhao, H. Larochelle, D. Englund, M. Solja˘ić, Deep learning with coherent nanophotonic circuits. *Nat. Photonics* **11**, 441–446 (2017).
7. R. Hamerly, L. Bernstein, A. Sludds, M. Solja˘ić, D. Englund, Large-scale optical neural networks based on photoelectric multiplication. *Phys. Rev. X* **9**, 021032 (2019).
8. J. C. Mikkelsen, W. D. Sacher, J. K. S. Poon, Dimensional variation tolerant silicon-on-insulator directional couplers. *Opt. Express* **22**, 3145–3150 (2014).
9. M. Kawachi, M. Okuno, K. Kato, K. Katoh, Y. Ohmori, A. Himeno, Silica-based optical-matrix switch with intersecting Mach-Zehnder waveguides for larger fabrication tolerances. *Conference on Optical Fiber Communication/International Conference on Integrated Optics and Optical Fiber Communication (1993), paper TuH4* (Optica Publishing Group, 1993), p. TuH4.
10. R. Nagase, A. Himeno, M. Okuno, K. Kato, K.-I. Yukimatsu, M. Kawachi, Silica-based 8×8 optical matrix switch module with hybrid integrated driving circuits and its system application. *J. Light. Technol.* **12**, 1631–1639 (1994).
11. T. Goh, A. Himeno, M. Okuno, H. Takahashi, K. Hattori, High-extinction ratio and low-loss silica-based 8 × 8 strictly nonblocking thermooptic matrix switch. *J. Light. Technol.* **17**, 1192–1199 (1999).
12. M. Okuno, K. Kato, R. Nagase, A. Himeno, Y. Ohmori, M. Kawachi, Silica-based 8 × 8 optical matrix switch integrating new switching units with large fabrication tolerance. *J. Light. Technol.* **17**, 771–781 (1999).
13. Y. Shoji, K. Kintaka, S. Suda, H. Kawashima, T. Hasama, H. Ishikawa, Low-crosstalk 2 × 2 thermo-optic switch with silicon wire waveguides. *Opt. Express* **18**, 9071–9075 (2010).
14. N. C. Harris, G. R. Steinbrecher, M. Prabhu, Y. Lahini, J. Mower, D. Bunandar, C. Chen, F. N. C. Wong, T. Baehr-Jones, M. Hochberg, S. Lloyd, D. Englund, Quantum transport simulations in a programmable nanophotonic processor. *Nat. Photonics* **11**, 447–452 (2017).
15. P. Dumais, D. J. Goodwill, D. Celo, J. Jiang, C. Zhang, F. Zhao, X. Tu, C. Zhang, S. Yan, J. He, M. Li, W. Liu, Y. Wei, D. Geng, H. Mehrvar, E. Bernier, Silicon photonic switch subsystem with 900 monolithically integrated calibration photodiodes and 64-fiber package. *J. Light. Technol.* **36**, 233–238 (2018).
16. K. Suzuki, R. Konoike, J. Hasegawa, S. Suda, H. Matsuura, K. Ikeda, S. Namiki, H. Kawashima, Low-insertion-loss and power-efficient 32 × 32 silicon photonics switch with extremely high-Δ silica PLC connector. *J. Light. Technol.* **37**, 116–122 (2019).
17. C. M. Wilkes, X. Qiang, J. Wang, R. Santagati, S. Paesani, X. Zhou, D. A. B. Miller, G. D. Marshall, M. G. Thompson, J. L. O'Brien, 60 dB high-extinction auto-configured Mach-Zehnder interferometer. *Opt. Lett.* **41**, 5318–5321 (2016).
18. M. Dong, G. Clark, A. J. Leenheer, M. Zimmermann, D. Dominguez, A. J. Menssen, D. Heim, G. Gilbert, D. Englund, M. Eichenfield, High-speed programmable photonic circuits in a cryogenically compatible, visible-near-infrared 200 mm CMOS architecture. *Nat. Photonics* **16**, 59–65 (2022).
19. K. Suzuki, G. Cong, K. Tanizawa, S.-H. Kim, K. Ikeda, S. Namiki, H. Kawashima, Ultra-high-extinction-ratio 2 × 2 silicon optical switch with variable splitter. *Opt. Express* **23**, 9086–9092 (2015).
20. M. Y.-S. Fang, S. Manipatruni, C. Wierzynski, A. Khosrowshahi, M. R. DeWeese, Design of optical neural networks with component imprecisions. *Opt. Express* **27**, 14009–14029 (2019).
21. R. Burgwal, W. R. Clements, D. H. Smith, J. C. Gates, W. S. Kolthammer, J. J. Renema, I. A. Walmsley, Using an imperfect photonic network to implement random unitaries. *Opt. Express* **25**, 28236–28245 (2017).
22. J. Mower, N. C. Harris, G. R. Steinbrecher, Y. Lahini, D. Englund, High-fidelity quantum state evolution in imperfect photonic integrated circuits. *Phys. Rev. A* **92**, 032322 (2015).
23. D. P. López, Programmable integrated silicon photonics waveguide meshes: Optimized designs and control algorithms. *IEEE J. Sel. Top. Quantum Electronics* **26**, 1–12 (2020).

24. A. López, D. Pérez, P. DasMahapatra, J. Capmany, Auto-routing algorithm for field-programmable photonic gate arrays. *Opt. Express* **28**, 737–752 (2020).
25. D. Pérez-López, A. López, P. DasMahapatra, J. Capmany, Multipurpose self-configuration of programmable photonic circuits. *Nat. Commun.* **11**, 6359 (2020).
26. S. Pai, B. Bartlett, O. Solgaard, D. A. B. Miller, Matrix optimization on universal unitary photonic devices. *Phys. Rev. Appl.* **11**, 064044 (2019).
27. T. W. Hughes, M. Minkov, Y. Shi, S. Fan, Training of photonic neural networks through in situ backpropagation and gradient measurement. *Optica* **5**, 864–871 (2018).
28. S. Pai, Z. Sun, T. W. Hughes, T. Park, B. Bartlett, I. A. D. Williamson, M. Minkov, M. Milanizadeh, N. Abebe, F. Morichetti, A. Melloni, S. Fan, O. Solgaard, D. A. B. Miller, Experimentally realized in situ backpropagation for deep learning in photonic neural networks. *Science* **380**, 398–404 (2023).
29. D. A. B. Miller, Setting up meshes of interferometers - reversed local light interference method. *Opt. Express* **25**, 29233–29248 (2017).
30. S. Pai, I. A. D. Williamson, T. W. Hughes, M. Minkov, O. Solgaard, S. Fan, D. A. B. Miller, Parallel programming of an arbitrary feedforward photonic network. *IEEE J. Sel. Top. Quantum Electron.* **26**, 1–13 (2020).
31. S. Bandyopadhyay, R. Hamerly, D. Englund, Hardware error correction for programmable photonics. *Optica* **8**, 1247–1255 (2021).
32. R. Hamerly, S. Bandyopadhyay, D. Englund, Asymptotically fault-tolerant programmable photonics. *Nat. Commun.* **13**, 6831 (2022).
33. D. A. B. Miller, Perfect optics with imperfect components. *Optica* **2**, 747–750 (2015).
34. M. Orshansky, S. Nassif, D. Boning, *Design for Manufacturability and Statistical Design: A Constructive Approach* (Springer Science & Business Media, 2007).
35. J. Wang, S. Paesani, Y. Ding, R. Santagati, P. Skrzypczyk, A. Salavrakos, J. Tura, R. Augusiak, L. Man˘inska, D. Bacco, D. Bonneau, J. W. Silverstone, Q. Gong, A. Acín, K. Rottwitt, L. K. Oxenløwe, J. L. O'Brien, A. Laing, M. G. Thompson, Multidimensional quantum entanglement with large-scale integrated optics. *Science* **360**, 285–291 (2018).
36. X. Qiang, X. Zhou, J. Wang, C. M. Wilkes, T. Loke, S. O'Gara, L. Kling, G. D. Marshall, R. Santagati, T. C. Ralph, J. B. Wang, J. L. O'Brien, M. G. Thompson, J. C. F. Matthews, Large-scale silicon quantum photonics implementing arbitrary two-qubit processing. *Nat. Photonics* **12**, 534–539 (2018).
37. C. Sparrow, E. Martín-López, N. Maraviglia, A. Neville, C. Harrold, J. Carolan, Y. N. Joglekar, T. Hashimoto, N. Matsuda, J. L. O'Brien, D. P. Tew, A. Laing, Simulating the vibrational quantum dynamics of molecules using photonics. *Nature* **557**, 660–667 (2018).
38. J. Carolan, C. Harrold, C. Sparrow, E. Martín-López, N. J. Russell, J. W. Silverstone, P. J. Shadbolt, N. Matsuda, M. Oguma, M. Itoh, G. D. Marshall, M. G. Thompson, J. C. F. Matthews, T. Hashimoto, J. L. O'Brien, A. Laing, Universal linear optics. *Science* **349**, 711–716 (2015).
39. A. Annoni, E. Guglielmi, M. Carminati, G. Ferrari, M. Sampietro, D. A. Miller, A. Melloni, F. Morichetti, Unscrambling light–Automatically undoing strong mixing between modes. *Light: Sci. Appl.* **6**, e17110 (2017).
40. A. Ribeiro, A. Ruocco, L. Vanacker, W. Bogaerts, Demonstration of a 4 × 4-port universal linear circuit. *Optica* **3**, 1348–1357 (2016).
41. M. Milanizadeh, P. Borga, F. Morichetti, D. Miller, A. Melloni, Manipulating free-space optical beams with a silicon photonic mesh. *2019 IEEE Photonics Society Summer Topical Meeting Series (SUM)* (IEEE, 2019), pp. 1–2.
42. L. Zhuang, C. G. H. Roeloffzen, M. Hoekman, K.-J. Boller, A. J. Lowery, Programmable photonic signal processor chip for radiofrequency applications. *Optica* **2**, 854–859 (2015).
43. J. Notaros, J. Mower, M. Heuck, C. Lupo, N. C. Harris, G. R. Steinbrecher, D. Bunandar, T. Baehr-Jones, M. Hochberg, S. Lloyd, D. Englund, Programmable dispersion on a photonic integrated circuit for classical and quantum applications. *Opt. Express* **25**, 21275–21285 (2017).
44. M. Prabhu, C. Roques-Carmes, Y. Shen, N. Harris, L. Jing, J. Carolan, R. Hamerly, T. Baehr-Jones, M. Hochberg, V. Čeperić, J. D. Joannopoulos, D. R. Englund, M. Solja˘ić, Accelerating recurrent Ising machines in photonic integrated circuits. *Optica* **7**, 551–558 (2020).
45. M. Reck, A. Zeilinger, H. J. Bernstein, P. Bertani, Experimental realization of any discrete unitary operator. *Phys. Rev. Lett.* **73**, 58–61 (1994).
46. W. R. Clements, P. C. Humphreys, B. J. Metcalf, W. S. Kolthammer, I. A. Walmsley, Optimal design for universal multiport interferometers. *Optica* **3**, 1460–1465 (2016).
47. R. Hamerly, S. Bandyopadhyay, D. Englund, Accurate self-configuration of rectangular multiport interferometers. *Phys. Rev. Appl.* **18**, 024019 (2022).
48. R. Hamerly, S. Bandyopadhyay, D. Englund, Stability of self-configuring large multiport interferometers. *Phys. Rev. Appl.* **18**, 024018 (2022).
49. D. A. Miller, Self-configuring universal linear optical component. *Photonics Res.* **1**, 1–15 (2013).
50. D. A. B. Miller, Self-aligning universal beam coupler. *Opt. Express* **21**, 6360–6370 (2013).
51. S. Grillanda, M. Carminati, F. Morichetti, P. Ciccarella, A. Annoni, G. Ferrari, M. Strain, M. Sorel, M. Sampietro, A. Melloni, Non-invasive monitoring and control in silicon photonics using CMOS integrated electronics. *Optica* **1**, 129–136 (2014).
52. L. G. Wright, T. Onodera, M. M. Stein, T. Wang, D. T. Schachter, Z. Hu, P. L. McMahon, Deep physical neural networks trained with backpropagation. *Nature* **601**, 549–555 (2022).
53. S. Bandyopadhyay, A. Sludds, S. Krastanov, R. Hamerly, N. Harris, D. Bunandar, M. Streshinsky, M. Hochberg, D. Englund, Single chip photonic deep neural network with accelerated training. arXiv:2208.01623 (2022). https://doi.org/10.48550/arXiv.2208.01623.
54. R. Hamerly, Meshes: Tools for modeling photonic beamsplitter mesh networks (2021); https://github.com/QPG-MIT/meshes.
55. Y. LeCun, C. Cortes, C. Burges, Mnist handwritten digit database. ATT labs (2010).
56. H. Xiao, K. Rasul, R. Vollgraf, Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. arXiv:1708.07747 (2017). https://doi.org/10.48550/arXiv.1708.07747.
57. T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, D. Ha, Deep learning for classical japanese literature. arXiv:1812.01718 (9999). https://doi.org/10.48550/arXiv.1812.01718.
58. Q. Wilmart, S. Brision, J.-M. Hartmann, A. Myko, K. Ribaud, C. Petit-Etienne, L. Youssef, D. Fowler, B. Charbonnier, C. Sciancalepore, E. Pargon, S. Bernabé, B. Szelag, A complete Si photonics platform embedding ultra-low loss waveguides for O- and C-band. *J. Light. Technol.* **39**, 532–538 (2021).
59. I. A. D. Williamson, T. W. Hughes, M. Minkov, B. Bartlett, S. Pai, S. Fan, Reprogrammable electro-optic nonlinear activation functions for optical neural networks. *IEEE J. Sel. Top. Quantum Electron.* **26**, 1–12 (2020).

Acknowledgments: We acknowledge discussions with S. Bandyopadhyay and A. Sludds. We thank NVIDIA Corporation for donating the Tesla K40 GPU used in this work. Calculations were also performed on the Engaging computing cluster at MIT. **Funding:** S.K.V. was supported by NSF RAISE-TAQS program, grant number 1936314, and the NTT Research Inc. grants "Large-scale nanophotonic circuits for neuromorphic computing" and "Netcast," administered by MIT. D.E. acknowledges partial support from programs NSF RAISE-TAQS (grant number 1936314) and NSF C-Accel (grant number 2040695). **Author contributions:** S.K.V: methodology, software, validation, investigation, visualization, writing—original draft, review, and editing. D.E.: conceptualization, visualization, and writing—review and editing. R.H.: methodology (primary), visualization, and writing—review and editing. **Competing interests:** D.E. is a scientific advisor to and owns stock in LightMatter Inc. D.E. serves as a part-time technical staff to Brookhaven National Laboratory with a focus on quantum computing and quantum networks (not related to this work). The other authors declare that they have no competing interests. **Data and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Materials. All the codes and trained models are available at Github (https://github.com/QPG-MIT/Maximally-Error-Tolerant-Mesh-ONNs.git) and Zenodo (https://doi.org/10.5281/zenodo.7943326). Finer details of the model transfer procedure are provided in the Supplementary Materials.

Submitted 26 February 2023
Accepted 12 June 2023
Published 12 July 2023
10.1126/sciadv.adh3436

Science Advances

Transferable learning on analog hardware

Sri Krishna Vadlamani, Dirk Englund, and Ryan Hamerly

Sci. Adv., **9** (28), eadh3436.
DOI: 10.1126/sciadv.adh3436

Science Advances (ISSN) is published by the American Association for the Advancement of Science. 1200 New York Avenue NW, Washington, DC 20005. The title *Science Advances* is a registered trademark of AAAS.