End-to-end optical backpropagation for training neural networks

Xianxin Guo,1,2,3, Thomas D. Barrett,2, Zhiming M. Wang,1 and A. I. Lvovsky2,4

1. Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan 610054, China
2. University of Oxford, Clarendon Laboratory, Parks Road, Oxford OX1 3PU, UK
3. Institute for Quantum Science and Technology, University of Calgary, Calgary, Canada, T2N 1N4
4. Russian Quantum Center, Skolkovo, 143025, Moscow, Russia

We propose the first practical scheme for end-to-end optical backpropagation in neural networks. Using saturable absorption for the nonlinear units, we find that the backward propagating gradients required to train the network can be approximated in a surprisingly simple pump-probe scheme that requires only passive optical elements. Simulations show that, with readily obtainable optical depths, our approach can achieve equivalent performance to state-of-the-art computational networks on image classification benchmarks, even in deep networks with multiple sequential gradient approximations. With backpropagation through nonlinear units an outstanding challenge to the field, this work provides a feasible path towards truly all-optical neural networks.

Machine learning (ML) is changing the way in which we approach complex tasks, with applications ranging from natural language processing to artificial intelligence and fundamental science. At the heart (or ‘brain’) of this revolution are artificial neural networks (ANNs), which are universal function approximators capable, in principle, of representing an arbitrary mapping of inputs to outputs. ANNs have multiple layers of nonlinear neurons, with the state encoded in one layer mapped to the next by multiplication with some weight matrix, which is a computationally expensive operation. This operation can, however, be readily implemented by leveraging the coherence and superposition properties of linear optics. Optics is therefore an attractive platform for realizing the next generation of neural networks, offering faster computation with low power consumption. Such proposals have been around for over thirty years and optical neural networks (ONNs) have been realized in both free-space and integrated settings. However, current ONNs are trained with, or heavily aided by, digital computers because an end-to-end optical implementation of the backpropagation algorithm required to train these networks has remained elusive. As a result, the computational advantages carried by optics remain largely unexploited. Here, we address this challenge and present a practical training method that allows ONNs to fully realise their potential.

The backpropagation algorithm aims to minimise a loss function that quantifies the divergence of the network’s current performance from the ideal, via stochastic gradient descent (SGD). To do so, the following steps are repeated until convergence: (1) forward propagation of information through the network; (2) evaluation of the loss function gradients with respect to the network parameters at the output layer; (3) backpropagation of these gradients to all previous layers; (4) parameter updates in the direction that maximally reduces the loss function. Forward propagation (step (1)) requires both the aforementioned matrix multiplication, to map information between layers, and a suitable nonlinear activation function applied individually to each neuron. Whilst this nonlinearity has so far been mostly applied digitally in hybrid optical-electronic systems – at the cost of repeatedly measuring and generating the optical state – recent work has also realized optical nonlinearities.

However, obtaining and backpropagating the loss-function gradients (steps (2-3)) remains an outstanding challenge in an optical setting. This is because the derivatives of the forward-propagating nonlinear response must be retained and then applied to the backwards-propagating signal. In 1987, Wagner et al. suggested that optical backpropagation could be achieved with approximated derivatives, however, this and other optical backpropagation proposals were never realized, or even analyzed in detail, largely due to their inherent complexity.

Here, we consider an optical nonlinearity based on saturable absorption (SA) and show that, with the forward-propagating features and the backward-propagating error taking the roles of pump and probe respectively, end-to-end backpropagation through the entire network can be realized using only passive optical elements. Our method is effective and surprisingly simple – with the required optical operations for both forwards and backwards propagation realised using the same physical elements. Simulations with physically realistic parameters show that the proposed scheme can train networks to performance levels equivalent to state-of-the-art ANNs. Moreover, learning is robust to deviations of the optically obtained training signal from the theoretical ideal. When combined with optical calculation of the error term at the output layer via interference, this completes a truly all-optical scheme for training an ONN.
RESULTS

Implementing optical backpropagation

We begin by recapping the operation of a neural network. Seeded with data at the input layer \((a^{(l)})\), forward-propagation maps the neuron activations from layer \(l-1\) to the neuron inputs at layer \(l\) as

\[
z_j^{(l)} = \sum_i w_{ji}^{(l)} a_i^{(l-1)},
\]

via a weight matrix \(w_{ji}^{(l)}\), before applying a nonlinear activation function individually to each neuron, \(a_j^{(l)} = g(z_j^{(l)})\) (with subscripts label individual neurons).

At the output layer we evaluate the loss function, \(\mathcal{L}\), and calculate its gradient with respect to the weights,

\[
\frac{\partial \mathcal{L}}{\partial w_{ji}^{(l)}} = \frac{\partial \mathcal{L}}{\partial z_j^{(l)}} \frac{\partial z_j^{(l)}}{\partial w_{ji}^{(l)}} = \delta_j^{(l)} a_i^{(l-1)},
\]

where \(\delta_j^{(l)} \equiv \partial \mathcal{L}/\partial z_j^{(l)}\) is commonly referred to as the ‘error’ at the \(j\)th neuron in the \(l\)th layer. From the chain rule we have

\[
\delta_j^{(l)} = \sum_k \frac{\partial \mathcal{L}}{\partial z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial z_j^{(l)}} = g'(z_j^{(l)}) \sum_k \delta_k^{(l+1)} w_{kj}^{(l+1)}.
\]

Given the errors at the output layer, i.e. \(\delta_j^{(L)}\), which is calculated directly from the loss function, we sequentially find the errors \(\delta_j^{(L-1)}, \ldots, \delta_j^{(1)}\) for all preceding layers using Eq. (3). These errors, as well as the activations \(a_i^{(l-1)}\) of all neurons, allow us to find the gradients \(\delta_j^{(l)}\) of the error function with respect to all the weights, and hence apply SGD.

The transformation (1) is readily implemented as a linear optical (interferometric) operation. Remarkably, the backpropagation (3) involves the same weight matrix, meaning that it can be implemented by physical backward propagation of an optical signal through the same linear optical arrangement (10). This reversible nature of linear optical transformations can be exploited in both free-space and integrated settings. To obtain the gradient (2) of the error function with respect to the weights, it suffices to tap off the forward- and backward-propagating light towards photodetectors, which measure, respectively, the activation and error values.

However, the above discussion has not yet covered the realisation of the activation unit \(g(z_j^{(l)})\). This unit must have the following features: (i) nonlinear response for the forward input; (ii) linear response for the backward input; (iii) modulation of backward input with the derivative of the nonlinear function. While it is natural to invoke nonlinear optics for this purpose, no design so far has been able to satisfy the requirement that the unit must respond differently to the forward and backward propagating light. Here, we show that this problem can be addressed using saturable absorption in the well-known pump-probe configuration.

Consider passing a strong pump, \(E_p\), and a weak probe, \(E_{pr}\), through a two-level medium (e.g. atomic vapour). The pump transmission is then a nonlinear function of the input,

\[
E_{P,\text{out}} = g(E_{P,\text{in}}) = \exp\left(-\frac{\alpha_0/2}{1 + E_{P,\text{in}}^2}\right) E_{P,\text{in}},
\]

where \(\alpha_0\) is the resonant optical depth and all fields are assumed to be normalised by the saturation threshold. A suitably weak probe, on the other hand, does not modify the transmissivity of the atomic medium, and hence experiences linear absorption with the absorption coefficient determined by the pump:

\[
\frac{E_{P,\text{out}}}{E_{P,\text{in}}} = \exp\left(-\frac{\alpha_0/2}{1 + E_{P,\text{in}}^2}\right).
\]

Note that both beams are assumed to be resonant with the atomic transition and therefore, as the phase of the electric field is unchanged, we treat these as real-valued without a loss of generality. Therefore, with the pump and probe taking the roles of forward-propagating signal and backward-propagating gradients in an ONN, required features (i) and (ii) of our optical nonlinear unit are satisfied.

Condition (iii) however remains to be satisfied. The
The key step that allows our scheme to realise all-optical backpropagation is to approximate this derivative with only the exponential transmission term, i.e. neglecting the square-bracketed factor in Eq. (6). By doing so, and recalling Eq. (4), we find that the backwards-propagating probe is automatically multiplied by this approximated gradient \( g'(E_{\text{p, in}}) \), exactly as is required to satisfy feature (iii). This may appear a coarse approximation, however any global scaling of the network gradients can be absorbed into the learning rate. Furthermore, as we will see, our approximation is only required to hold within the nonlinear region of the SA response. The efficacy of this approximation will be systematically investigated in a later section.

For concreteness, Fig. 1(a) shows how such an optical backpropagation scheme could be practically realised for a single network layer, where, for illustrative purposes, we consider a free-space implementation of matrix multiplication using a spatial light modulator (SLM) [20]. Regardless of the chosen platform, passive optical elements can only implement weighted connections that satisfy conservation of energy. However, for a single hidden layer, this is not a practical limitation as such weight matrices can be effectively realised with normalised weights by correspondingly rescaling the neuron activations in the input or output layer. For deep networks with multiple layers, the absorption through the vapour cell will reduce the field amplitude available to subsequent layers. This can be counteracted by inter-layer amplification.

The remaining, not yet discussed, element of the ONN training is the measurement and re-injection of the error \( \delta^{(L)} \) at the output layer, to initiate backpropagation. To implement this optically, we train the ONN with the mean-squared-error loss function,

\[
\mathcal{L} = \sum_{i} \frac{1}{2} \left( z_{i}^{(L)} - t_i \right)^2 ,
\]

where \( t_i \) is the target value for the \( i \)th output neuron. This error function implies \( \delta^{(L)} = \partial \mathcal{L} / \partial z_i^{(L)} = z_i^{(L)} - t_i \), which is calculable by interference of the network outputs with the target outputs on a balanced beam-splitter. This approach to error calculation is illustrated in the right panel of Fig. 1(b), whereas the left panel shows the standard approach in which the errors are calculated offline (electronically).

With the network configuration described above, the only parts that require electronics are (a) measurements of the errors in each layer, (b) generating the training inputs and target reference beams and (c) updating the weights. In practice, the update (c) is calculated not for each individual training set element, but as average for multiple elements (a “mini-batch”), hence the speed of this operation is not critical for the ONN performance. Generating the inputs and targets (b) is decoupled from the calculation performed by the ONN and requires fast optical modulators, which are abundant on the market. Finally, the measurements (a) must be followed by calculating the product \( \delta^{(L)} a_i^{(L-1)} \) and averaging over the mini-batch. This simple operation can be implemented using electronic gate arrays. Alternatively, the multiplication can be realised by homodyne detection (interference with a local oscillator followed by intensity detection) and analog averaging using boxcar integrators.

**Case study: image classification**

To examine the efficacy of our proposed scheme, we train simulated ONNs to solve the canonical ML task of image classification. In order to obtain a comparison benchmark, we also computationally train ANNs with the same architecture using standard best practices. Concretely, for ANNs we use ReLU (rectified linear unit) activation functions, defined as \( g_{\text{ReLU}}(z) = \max(0, z) \), and the categorical cross-entropy loss function, which is defined as \( \mathcal{L} = - \sum_{i} t_i \log(p_i) \) where \( p_i = \exp(z_i^{(L)}) / \sum_k \exp(z_k^{(L)}) \) is the softmax probability distribution of the network output. We hereafter refer to these two types of network as optical and benchmark networks, respectively.

Our first set of numerical experiments is to classify images of handwritten digits from 0 to 9 from the MNIST [21] dataset containing greyscale bitmaps of size \( 28 \times 28 \), which are fed into the input layer of the ONN. The output layer contains \( 10 \) neurons whose target values are 0 or 1 dependent on the digit encoded in the bitmap (“one-hot encoding”). We use a network architecture with two 128-neuron hidden layers as shown in Fig. 2(a)(i). Further details of the network and training can be found in the Methods.

Fig. 2(a)(ii) compares the simulated performance of the optical and benchmark networks. The ReLU-based classifier achieves an accuracy of \( (98.0 \pm 0.2) \% \), which provides an approximate upper bound on the achievable performance of this network architecture for the chosen task [22]. An optical network with an optical depth of \( \alpha_0 = 30 \) exactly matches this level of performance with a \( (98.0 \pm 0.2) \% \) classification accuracy. As an additional benchmark, we train the optical network using the exact gradient, Eq. (6), of the activation function, obtaining a similar accuracy of \( (98.1 \pm 0.3) \% \). The convergence speed to near-optimun performance during training is unchanged across all of these networks.

Fig. 2(a)(iii) shows the trained performance of opti-
To probe the limits of the achievable performance using SA nonlinearities and optical backpropagation, we now also consider the more challenging Kuzushiji-MNIST (KMNIST) and Extended-MNIST (EMNIST) datasets. For these applications we use a deep network architecture with convolutional layers (see Methods for details), as illustrated in Fig. 2(b)(i), which significantly increases the achievable classification accuracy to near optimal levels once $\alpha_0 = 10$, which is readily obtainable experimentally.

To examine approximation errors

With both approximated and exact gradients producing near equivalent performance on image classification, it is natural to consider why this is the case and when it may break down. To this end, we simplify our simulated ONN to contain only a single 128-unit hidden layer, and train it for MNIST recognition. From Fig. 3(a) we can see that during training the neurons are primarily distributed in the unsaturated region, (i), of the activation function. This is a consequence of the fact that the expressive capacity of neural networks arises from the nonlinearity of its neurons.

Therefore, to train the network, the optically-obtained gradients need to approximate the exact gradients (up to a fixed scaling as previously discussed) in only this nonlinear region. From Fig. 3(b) we can see that this is the case for a SA nonlinearity with $\alpha_0 = 10$.

It is interesting to investigate how imprecision in the calculated derivatives affects training. To this end, we...
evaluate the network performance by replacing the gradient \( g'(\cdot) \) with random functions of varying similarity to the true gradient, Eq. \( \text{(6)} \), within the nonlinear region (the quantitative measure, \( S_i \) of the similarity is defined in the Methods). From Fig. 3(c) we see that the performance appears robust to approximation errors, defined as \( 1 - S_i \), of up to \( \sim 15\% \). We explain this potentially surprising observation by noting that SGD will converge even if the update vector deviates from the direction towards the exact minimum of the loss function, so long as this deviation is not too significant.

In the case of SA, i.e. when the approximate gradients given by Eq. \( \text{(5)} \) are used, this error saturates at \( \sim 10\% \) for increasing optical depth, see Fig. 3(d), so no significant detrimental effect on the training accuracy can be expected. These results suggest that our scheme would still be effective in a noisy experimental setting and that the approach studied here may function well for a broad range of optical nonlinearities.

**DISCUSSION**

In conclusion, we have designed the first practical end-to-end optical implementation of the backpropagation algorithm. Such an optically trained ONN offers considerable advantages with respect to computational ANNs. We expect the energy cost per floating point operation (FLOPs) to be orders of magnitude lower than current GPU implementations. Moreover, the speed of optical matrix multiplication is fundamentally determined by the speed of light propagating through the system, and currently available hardware already allows the network to be trained at least as fast as an electronic system. (Further details on these points can be found in the Methods.)

We anticipate that similar concepts could be successfully applied to any nonlinearity for which a suitable proxy for the response function gradient can be obtained. Therefore, as well as presenting a truly all-optical neural network, this work sets out an important consideration for nonlinearities in the design of analog neural networks of any nature.

**ACKNOWLEDGEMENTS**

A.L.’s research is partially supported by Russian Science Foundation (19-71-10092). X.G. acknowledges funding from the University of Electronic Science and Technology of China. A.L. thanks William Andregg for introducing him to ONNs.

**AUTHOR CONTRIBUTIONS**

The research was conceived by X.G. and A.L. X.G. proposed the idea of using SA to implement the activation function. T.D.B. and X.G. jointly performed the simulations. The manuscript was prepared by T.D.B., X.G. and A.L. All work was done under the supervision of A.L. and Z.M.W.
We consider three different datasets, all containing 28×28 pixel greyscale images: MNIST [21], Kuzushiji-MNIST [23] (KMNIST) and Extended-MNIST [24] (EMNIST). MNIST corresponds to handwritten digits from 0 to 9, KMNIST contains 10 classes of handwritten Japanese cursive characters, and we use the EMNIST Balanced dataset, which contains 47 classes of handwritten digits and letters. MNIST and KMNIST have 70,000 images in total, split into 60,000 training and 10,000 test instances. EMNIST has 131,600 images, with 112,800 (18,800) training (test) instances. For all datasets, the training and testing sets have all classes equally represented.

**Network architectures**

The fully-connected network we train to classify MNIST (corresponding to the results in Fig. 2(a)) first unrolls each image into a 784-dimensional input vector, before a single 128-neuron hidden layer and a 10-neuron output layer.

The convolutional network depicted in Fig. 2(b)(i) has two convolutional layers of 32-channel and 64-channels, respectively. Each layer convolves the input with 5×5 filters (with a stride of 1 and no padding), followed by a nonlinear activation function and finally a pooling operation (with both kernel size and stride of 2). After the convolutional network, classification is carried out by a fully-connected network with a single 128-neuron hidden layer and NC-neuron output layer, where NC is the number of classes in the target dataset.

**Network training**

All networks are trained with a mini-batch size of 64 and a fixed learning rate of 5×10⁻⁴. For each network, the test images of the target dataset are split evenly into a ‘validation’ and ‘test’ set. After each epoch, the performance of the network is evaluated on the held-out ‘validation’ images. The best ONN parameters found over training are then used to verify the performance on the ‘test’ set. The fully-connected networks were trained on MNIST for 50 epochs. The convolutional networks are trained for 20 epochs when using ReLU, Tanh or Sigmoid nonlinearities, and 40 epochs when using SA nonlinearities.

Training performance is empirically observed to be sensitive to the initialisation of the weights, which we ascribe to the small gradients away from the nonlinear region of the SA response curve. For low optical depths, ω₀ ≤ 30, all layers are initialised as a normal distribution of width 0.1 centred around 0. For higher optical depths, the weights of the fully-connected ONN shown in Fig. 2(a) are initialised to a double-peaked distribution comprised of two normal distributions of width 0.15 centred at ±0.15.

For all images, the input is rescaled to be between 0 and 1 (which practically would correspond to 0 ≤ E_p(0) ≤ 1) when passing it to an network with computational nonlinearities (i.e. ReLU, Sigmoid or Tanh).

Due to ‘absorption’ in networks with SA nonlinearities, we empirically observe that rescaling the input data to higher values results in faster convergence when training convolutional networks with multiple hidden layers. Therefore, the fully connected networks in Fig. 2(a) use inputs between 0 and 1 and the convolutional networks in Fig. 2(b) use inputs normalised between 0 and 5(15) for ω₀ ≤ 10(ω₀ > 10).

**Calculation of gradient approximation error**

As discussed in the main text, we approximate the true gradients g(·) of the activation functions by random functions f(·) to test the effect of the approximation error on training.
Here we discuss how these functions are generated and how the similarity measure is defined.

The response of a saturable absorption nonlinearity can be considered in two regimes, nonlinear (unsaturated) and linear (saturated), which are labelled (i) and (ii) in Fig. 3(b), respectively. During the network training, the neuron input values ($z^{(i)}$) are primarily distributed in the nonlinear region, as seen in Fig. 3(a) and discussed in the main text. Therefore, we model the neuron input as a Gaussian distribution within this region,

$$p(z) = \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{z^2}{2\sigma^2} \right),$$

where $2\sigma$ is the width of region (i). We then define the similarity as the reweighted normalised scalar product between the accurate and approximate derivatives,

$$S = \frac{\left| \int f(z)g'(z)p(z)\,dz \right|^2}{\int [f(z)]^2p(z)\,dz \cdot \int [g'(z)]^2p(z)\,dz}.$$  

According to the Cauchy-Schwarz inequality, $S$ is bounded by 1 and therefore so is the average approximation error, $1 - S$.

To obtain the results in Fig. 3(c-d), we generate 200 random functions for $f$, with different approximation errors. We first generate an array of pseudo-random numbers ranging from 0 to 1, concatenate it with the flipped array to make them symmetric like the derivative $g'(\cdot)$, and then use shape-preserving interpolation to obtain a smooth and symmetric random function. The network is then trained once with each of the generated $f$’s.

**Power consumption and speed of an ONN**

The power consumption and computation speed of an ONN are dependent on the network architecture and implementation details. Therefore, for concreteness, we now consider a fully-connected network with a 784-dimensional input (order 1000 optical modes), and a single 128-neuron hidden layer, with SA optical nonlinearities implemented on the $^{87}$Rb D$_2$ line.

Recalling Fig. 2(b), we note that during training the input power to each neuron is typically restricted to the unsaturated region, (i), of the nonlinearity response. For the SA nonlinearities we consider, the saturation intensity is given by

$$I_{\text{sat}} = \frac{\hbar\omega\Gamma}{2\sigma_0} = 16.6 \text{ pW mm}^{-2}, \quad (10)$$

where $\Gamma = 2\pi \times 6 \text{ MHz}$ is the natural linewidth, and $\sigma_0 = 3\lambda^2/(2\pi)$ is the resonant absorption cross section. For beams with a waist of $w_0 = 100 \mu m$, this corresponds to a saturation power of the $P_{\text{sat}} \approx 500 \text{nW per neuron}$, and therefore total input powers on the order of 50 pW to 500 pW would be expected.

With a reasonable pulse duration of 100 ns (which is longer than the excited state life time), the maximum energy cost of a single forward pass through the network is on the order of nJ, and the backpropagation energy cost is negligible. Therefore, the energy cost per floating point operation (FLOP) is on the order of fJ, which is several orders of magnitude better than GPUs and electronic computers.

The computation speed of the ONN is limited by the bandwidth of the modulator used to encode the input pattern and weight matrix. Considering the free space implementation shown in Fig. 1(a), the bandwidth’s of SLM’s typically range between 10 Hz and 10 kHz. When training an $L$-layer neural network with $N$ neurons in each layer, a GHz-speed electronic computer finishes the forward and backward pass in $\sim 4LN^2$ ns. For ONN’s to match this performance, the bandwidth of the spatial light modulator should be at least $1/4LN^2$ GHz. For a simple 2-layer network with $N^2 \sim 10^5$, the modulator bandwidth should be on the order of kHz, which can be readily achieved with a digital micromirror device. Even if one works with slow liquid-crystal spatial light modulator, computation speedup can be obtained by increasing the network size. Further dramatic speedup can be achieved using electro-optical modulators based on ferroelectric crystals, which operate on nanosecond time scales; this technology has recently been incorporated into integrated platforms.

**Code availability**

Source code has been made publicly available at https://github.com/tomdbar/all-optical-neural-networks.