Supercontinuum neural network and analog computing evaluation

Kevin F. Lee* and Martin E. Fermann
IMRA America Inc., 1044 Woodridge Ave, Ann Arbor, Michigan 48105, USA

(Received 2 August 2023; revised 16 January 2024; accepted 4 March 2024; published 20 March 2024)

We use octave-spanning, phase-shaped supercontinuum generation as an analog computing element in a neural network. We can perform standard machine learning tasks such as autoencoding by embedding the physical device within a virtual neural network which converts input data to optical parameters, and measured spectra to output data. To understand the computing potential of the supercontinuum, we fully measure a small subset of input parameter space. We test universal function approximation by forming a basis set of delta functions from linear combinations of the physically measured functions. This approach provides a more general way to estimate the computational power of a physical operation than performing a specific task.

DOI: 10.1103/PhysRevA.109.033521

I. INTRODUCTION

Electronic computing power has limits such as heat and memory access, but there is always demand for more. Analog or physical computing is being researched as a way to diversify beyond semiconductors to calculate with more efficiency or speed by making use of physical processes that are compatible with certain computation types [1]. The high bandwidth and efficiency of optics makes it a good candidate for augmenting electronic computing. Optics long ago replaced electronics for long-distance data transfer with its large bandwidth and efficient propagation. Adding more processing before handoff to electronic systems is a natural next step [2]. An optical computer data bus could boost memory and processor sharing in massively parallel computing [3]. Energy efficiency [4], parallelization, and multiplexing make optics a good candidate to take over application-specific computing tasks such as neural networks (see Refs. [5–8] for some reviews). The inexact nature of most neural network tasks makes them particularly compatible with the approximate nature of analog computing [9].

Optical computing usually uses weak nonlinearity such as squaring the field by photodetection, or low-order harmonic generation [1,10], since weaker nonlinearities are easier to understand and engineer. Stronger nonlinearity contributes to improved machine learning [11], and recently stronger nonlinearity has been used, such as picosecond pulses in multimode fibers [12], programmable resonators on chip [13], soliton propagation [14], and soliton fission [15]. In the soliton fission case, they output spectra with shifted peaks in wavelength, purposely restricting themselves to tenth-order solitons to avoid instability. The spectrum covers almost an octave, but this is a gradual shifting over a 100-m-long nonlinear fiber [15].

Here, we go to the strongest optical nonlinearity not involving material damage, supercontinuum generation in highly nonlinear fiber. The spectrum looks like a mess spread across an octave, but the spectra can be quite reproducible, and can be controlled by shaping the input pulse [16]. It is these richly detailed spectra that provide a wide variety of spectral output as a function of the laser pulse input.

Neural network computing works by offering an enormous variety of behaviors (that is, how the outputs respond to inputs) in a large parameter space, from which the user tries to find settings that will perform the desired task. Adding neurons multiplies the parameter space size and computing power (as well as difficulty searching the space). For a given optical system, higher optical nonlinearity is like adding neurons, providing many more behaviors for the user to choose from. For example, a supercontinuum system at low power can act like a soliton fission computer [15], but by making thousands of higher pump power settings available, we vastly increase the number of behaviors to work with.

In a hypothetically ideal optical computer, you would want a tool to couple beams with variable strength, or change a beam's wavelength at will, allowing you to calculate, process, and route signals. With the flexibility of highly nonlinear supercontinuum shaping, all these tasks could be performed in a single pass. An optical processor might be a large set of devices written onto a chip, such as neuromorphic interferometers [17]. A supercontinuum waveguide is a continuous processor where light interferes and nonlinearly changes itself, effectively performing many operations over the waveguide length. The behavior is controlled by the condition of the input light, meaning phase shaping can be applied at various stages before the supercontinuum.

We introduce three main ideas with supercontinuum computing:

(1) Highly nonlinear supercontinuum generation can reliably perform calculations. Here, we integrate the supercontinuum into a neural network using image autoencoding as an example. Trainable layers in a computer translate input data into pulse shaper settings, and another set of layers translates the resulting supercontinuum spectra into output data.

(2) Analog computers can be evaluated by comprehensively measuring small, manageable subsets of their input parameter space, and generating a complete delta function

*klee@imra.com

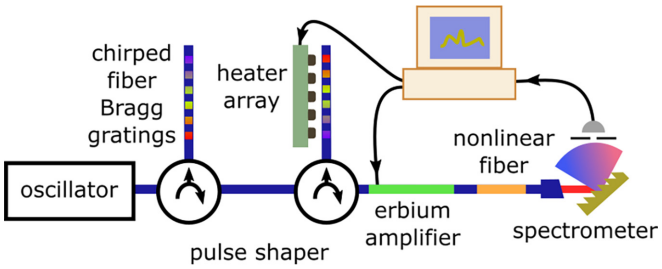

FIG. 1. System outline. Pulses from a femtosecond fiber oscillator are stretched and recompressed in oppositely oriented chirped fiber Bragg gratings. Heaters on one grating control group delay dispersion. Pulses are amplified to about 1 nJ in a fiber amplifier with computer-controlled pumping for supercontinuum generation in 41 cm of small-core nonlinear fiber. This length is much longer than needed for octave-spanning broadening when the pulse is well compressed. A computer controls the heaters, amplifier pump current, and records the supercontinuum spectrum measured by a spectrometer.

basis set to estimate the resolution to which universal function approximation is possible. This provides a recipe for determining the useful resolution of an analog system, and for ensuring universal approximation within the subset. Though limited by its brute force nature, this check operates at a complexity comparable to common demonstrations like classifying the modified NIST (MNIST) handwritten digit set, which can be solved quickly on a computer by very small artificial neural networks.

(3) A training method for incorporating analog devices into neural networks. After emulating the physical device, standard virtual training generates imperfect translation layers. The input translator and physical device convert the training data into measured spectra, and the output translator is then retrained on a computer with this physical data. Our method is good for slow, small parameter space systems, since once the emulation and training spectra datasets are acquired, the training and parameter optimization is fully virtual, instead of repeating open-ended live training sessions that discard data.

Comparisons of our hybrid supercontinuum neural network to computer-only calculations show that the supercontinuum improves the result. From this and our comprehensive parameter subspace measurements, we know that digital inputs shape the supercontinuum spectrum in complex but predictable and useful ways. This kind of spectral control would be an ideal element in an optical computer or data bus.

II. EXPERIMENTAL SYSTEM

The device, sketched in Fig. 1, is an erbium, polarization-maintaining, femtosecond fiber laser. The pulse shaper is a pair of identical chirped fiber Bragg gratings in opposite directions [16]. The pulse wavelengths reflect at different points along the 5-cm grating length, imparting a large chirp. The opposite directions means the chirp nominally cancels, but applying heat to the grating will change the wavelength reflected in that area. One grating is mounted on an array of 32 heaters, providing computer control of the group delay dispersion. The controller has 12-bit resolution, and the six heaters at the edges do not change the pulse much. We allow 20 s for settling after changing a heater.

The phase shaped pulse is amplified in erbium gain fiber with computer-controlled pump current, then sent through 41 cm of low-dispersion, small-core fiber for supercontinuum generation. Given the range of control from phase and intensity, the computer can generate spectra ranging from narrow to highly structured octave spanning. The spectrometer is an extended InGaAs scanning grating spectrometer which takes on the order of a second to return a spectrum with 6001 wavelengths.

From the computer's perspective, the physical system is a black box that accepts 33 inputs and returns 6001 outputs. Within an artificial neural network with conventional multilayer feedforward perceptrons, the physical system looks like a set of hidden layers with fixed weights. Integration into a neural network then only requires matching the number of parameters, so 33 neurons in the layer preceding the optics, and any number of neurons that accept 6001 inputs in the layer following the optics. We will call these translators, that translate the input data to actuator settings, and translate a spectrum into the desired output values.

We take a general approach which relies on machine learning to find the best way to use the physical device. Other approaches use fixed relations, such as directly mapping input values and trainable parameters to actuator settings [1], using physical outputs directly [18], or with simple calculations such as binning [15] for creating digital outputs.

The fixed approach is easier to understand, but is effectively the user manually creating some or all of the input and output translators, limiting the system to a small subset of the whole parameter space. An example of a bad choice would be assigning an input to a heater with little light at that wavelength, making that input effectively disappear. Switching tasks would also require the user to choose new mappings. Trainable translators automatically choose the mapping, ideally making good use of the full parameter space.

III. COMPUTING AS FUNCTIONS

Computing is evaluating a function. Even the generation of an image from a text prompt is taking an input and returning output values. A neural network approximates functions, where training tunes the network until it outputs satisfactory results. In analog computing, a physical device is a fixed function of its parameters. The power at each wavelength λ in the supercontinuum spectrum is a function of the actuator values $\vec{x}$: $P_\lambda(\vec{x})$. Our grating spectrometer returns powers at 6001 wavelengths, so the physical device is a parallel calculator of 6001 P_λ functions.

Figure 2 is a cartoon of the supercontinuum behaving as functions. We imagine varying only heater #15 with the rest off, and measuring spectra (columns) across its range of values. Going across a row at a particular wavelength gives the spectral intensity as a function of the heater value, such as $P_{1800\text{nm}}(x_{15})$. This function is unlikely to match the target function, but there are many more available, such as $P_{1600\text{nm}}(x_{15})$. In a conventional neural network, a neuron takes linear combinations of the layer below it, so weighted sums

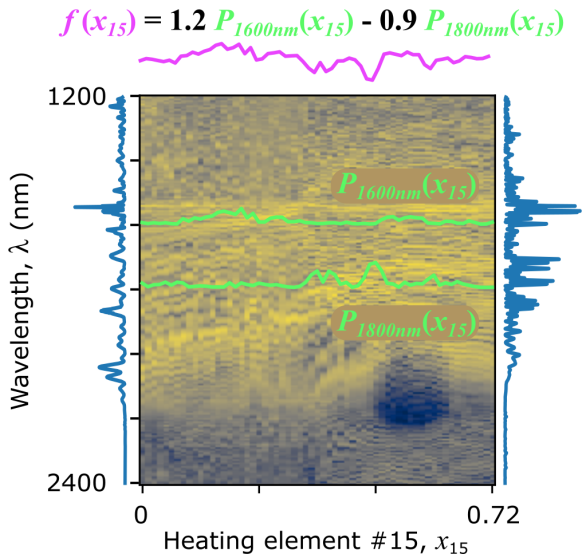

FIG. 2. Cartoon of creating functions from data. The color map is several spectra (columns) for a range of values of heater #15 only, reducing the vector of actuator settings $\vec{x}$ to one element, x_{15}. Spectra for $x_{15} = 0$ and 0.72 are illustrated vertically next to their columns. Each row is a function $P_\lambda(x_{15})$. Linear combination can make a new function f from the available P_λ such as $P_{1600\text{nm}}$ and $P_{1800\text{nm}}$ weighted to 1.2 and -0.9 (with the others weighted to zero). Evaluating $f(0.5)$ would be be measuring a spectrum with heater #15 set to 0.5, and summing $1.2\times$ the intensity at 1600 nm with $-0.9\times$ the intensity at 1800 nm. If P_λ has enough variety of shapes, f can approximate any function by choice of weights.

can be taken, such as the illustrated combination: $f(x_{15}) = 1.2P_{1600\text{nm}}(x_{15}) - 0.9P_{1800\text{nm}}(x_{15})$, providing an even greater range of available functions.

Writing this out for all heaters and including the customary offset, f is a linear combination of the available functions P_λ weighted by A_λ, and a bias b: $f(\vec{x}) = b + \sum_\lambda A_\lambda P_\lambda(\vec{x})$. If the P_λ's are varied enough to be a complete set, then f can be any arbitrary function, and the hybrid of a physical device and a virtual neuron is, like an artificial neural network [19], a universal approximator.

To show completeness, we want to generate all functions of a known complete set using linear combinations of P_λ. Well-known functions such as the Fourier series form a complete basis (given enough functions for the resolution). We choose a basis that is easy to understand, the set of multidimensional Dirac δ functions that return 1 for a particular input value, and 0 otherwise. Expansion coefficients of a function in the δ basis are simply the function value at each δ peak position. This is just like describing a grayscale image based on the brightness (coefficient) of each pixel (delta function peaked at that location).

A. One-dimensional basis

Our system controls one pump diode and 32 heaters at 12 bit resolution. If we limit ourselves to 20 actuators with 1000 levels each, there would still be 10^{60} possible input values and corresponding delta functions in a complete set. We cannot measure so many spectra, so we select subsets of the full input

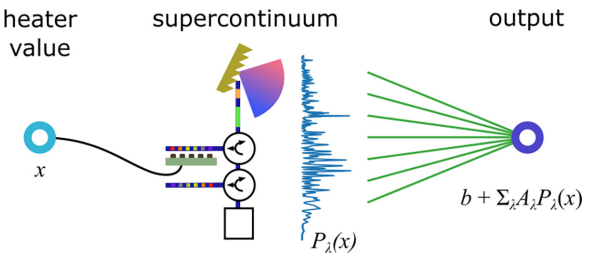

FIG. 3. Schematic of one-dimensional (1D) delta function arrangement. The input space is the range of one pulse-shaping heater. A single neuron reads the spectrum for a given heater setting, taking the weighted sum of the spectral elements to produce a single output value. For an example target function of $\delta(x - 0.2)$, the neuron parameters b and A_λ are trained to give an output of zero unless $x = 0.2$, in which case it should return 1. For the two-dimensional (2D) and three-dimensional (3D) sets, there are two or three input heater values, but the rest of the schematic is the same, with the output neuron converting a spectrum to a single value.

parameter space that are small enough to comprehensively measure. We start by stepping one heater at fairly high resolution from 0 to 0.6 (0: no heat, 1: full heat, which is about +20K for one heater) for 482 total points. Taking all these spectra yields all values of P_λ within this small subspace. We later permute two, then three heaters.

We perform fits to find the 482 sets of A_λ and b that approximate each of the 482 δ functions. The fits were performed by standard neural network training. As in Fig. 3, the actuator setting was the input; corresponding spectra were retrieved from the comprehensive measurement dataset; and a single output neuron with no activation function was trained.

The training data was first constructed with one data point for each possible input and all the corresponding outputs being 0 except for a 1 at the peak of the target δ function (e.g. [input, output]: [0,0], [0.1,0], [0.2,1],...). Given the nature of the delta function, the contribution of the peak to the error of the fit is small compared to the large number of points with outputs of zero. The fit then reduces the error by scaling down everything, accepting a lower peak of, say, 0.8 instead of 1 in exchange for a smaller baseline noise around output 0. To compensate for this, we added more copies of the data point with output 1 to the training data, effectively increasing its weight in the fit because it exposes the training routine to more instances of the peak.

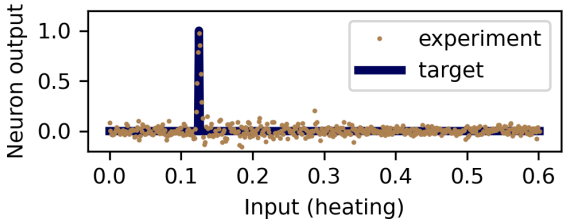

FIG. 4. Single δ function generated from supercontinuum spectra by a virtual neuron. The peak width gives an estimate of the input resolution, and the vertical point distribution around zero output gives an estimate of the system noise.

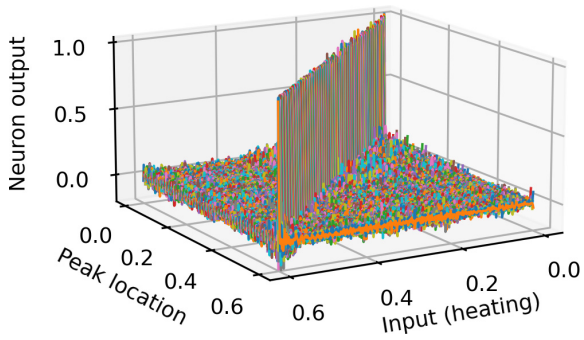

FIG. 5. δ functions from supercontinuum spectra for all 482 input values. Since a δ function can be formed at all points, the supercontinuum can approximate any function to a resolution of about 120 points in this parameter range.

Figure 4 shows a fit to a δ function. When we acquire data, we actually save two spectra per setting. One set is used to train the neuron. The second set is processed by the trained neuron. If the system acts like a clean δ function with the second set as inputs, then we know that the fit was successful and does not rely on noise from the first set.

The neuron generates a clear peak with a full-width at half-maximum (FWHM) of 0.005. Dividing the input range by the peak width gives an input axis with 120 resolvable points. A very rough approximation takes each δ function to be worth one layer of two neurons with activation (enough to emulate a δ function), or about 240 neurons in this case. In the best case for 20 actuators with 120 points, $120^{20} = 4 \times 10^{21}$ resolved points would be the upper limit. Away from the peak, the output values around zero have a distribution FWHM of 0.07 (thickness of horizontal line), which is a measure of the output noise.

All 482 δ functions are stacked along the peak location axis in Fig. 5. The generated δ functions are of similar quality to the sample in Fig. 4. This means that we can approximate any function up to a resolution of about 120 points ($\sim$7 bits) over 0–0.6 of the full heating range. We test this by fitting to an arbitrarily chosen function, a chirped sinusoid, shown in Fig. 6. It follows fast changes, suggesting all 482 points may be useful. This is a direct fit to the sinusoid, not a linear combination of the fitted delta functions, which would be noisier.

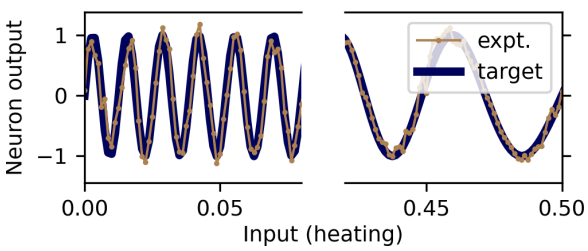

FIG. 6. Chirped sinusoid (thick line) approximated with a linear combination of measured supercontinuum spectra as a function of one heater element (points and thin line). The generated function closely follows the target.

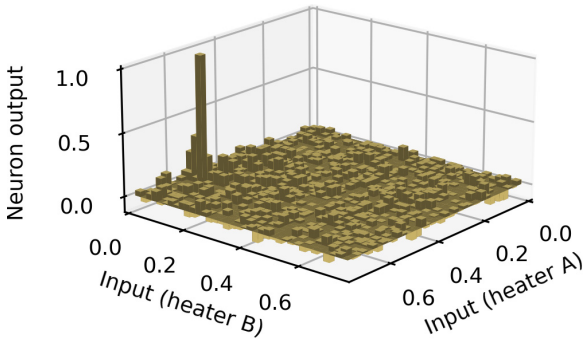

FIG. 7. Single δ function with inputs of two heater values. There is a clear peak and reasonable noise floor. The tested resolution is low to reduce acquisition time.

B. Two-dimensional basis

Next we permute two heaters. To save time, we greatly reduced the scan resolution to 37 points per heater, though this is still $37^2 = 1369$ possible input values. A fit to a single 2D δ function is shown in Fig. 7. There is a clear peak, although it is fairly wide. The background noise is comparable to the 1D case. Fits were performed for all δ functions to confirm clear peaks for all points. If we consider every other point to be resolvable, we have $18^2 = 324$, more than the 120 possible values we found for a single heater, but far below $120^2 = 14\,400$.

C. Three-dimensional basis

We also performed a limited 3D scan of 13 points within 0–0.06 of full range for $13^3 = 2197$ possible input values and spectra, making for a lengthy acquisition. Figure 8 shows one of the lower-quality emulated δ functions, which would ideally be one voxel (a 3D pixel) with value 1 in 3D space. The shapes for these fits are not very consistent, but in general the main voxel has value 1, and there are a number of adjacent voxels with significant values. Given the asymmetric shapes, if we loosely consider the width to be 0.02, then there might

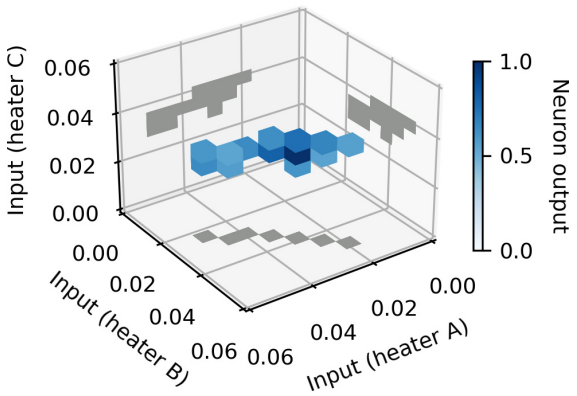

FIG. 8. A single δ function for three heaters, with voxels only plotted for output values over 0.5. This is one of the lower-quality instances, and the peak is larger and less consistent than the 1D and 2D cases, but it still forms a localized object that can act as a low-resolution δ function. Projections on the walls are visual aids.

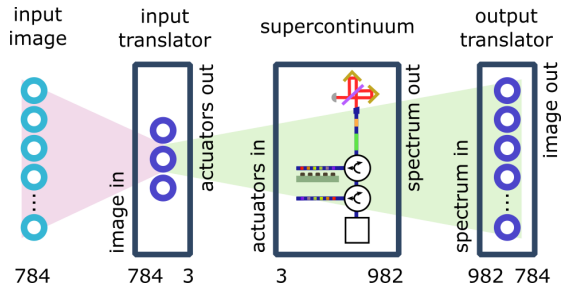

FIG. 9. Autoencoder schematic. The 784 grayscale pixels of a fashion MNIST image are read by three input translator neurons, each of which calculates an actuator setting for the physical device. These three values are the encoded version of the image. The physical system returns a supercontinuum spectrum of 982 elements. These are read by the 784 output translator neurons, each of which outputs a pixel in the decoded image. The input and output translator neurons are trained to ideally replicate the input image at the output.

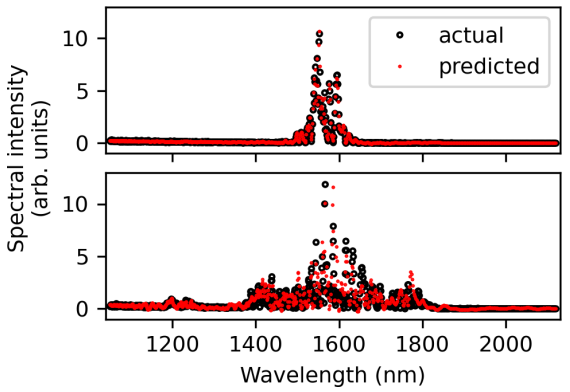

FIG. 10. Examples of a well-emulated (top) and poorly emulated spectrum (bottom) when testing the neural network made to emulate the supercontinuum. Details of specific peaks may be incorrect, but the overall shape is predicted well.

be 30 resolved points within 0–0.6, and $30^3 = 27\,000$ points for the three heaters.

With so many parameters, it is not clear whether the decreasing emulation quality is due to limitations of the supercontinuum or just poor fitting. Fitting to a δ function with 2197 points takes even longer than the 20-s wait time for the heaters to settle. Rather than continuing to add dimensions, this evaluation method could be extended by sampling more varieties of input subspaces, such as random vectors within the full input space to provide greater confidence in the usefulness of the full input space.

These tests provide a baseline for evaluating our supercontinuum, and analog computers in general. The 1D test indicates hundreds of resolvable input points, which provides a clear upper limit to available input points. The two- and three-heater tests are progressively less clear, and do not show multiplicative increase in points when adding input dimensions. This may be an issue with fit quality, but similar problems will occur when using it in a neural network calculation. It is clear, though, that our hybrid supercontinuum network is useful for general purpose approximation within noise limits with at least hundreds of input points.

IV. APPLICATION TO AUTOENCODING

We now demonstrate the standard machine learning task of autoencoding images, where an image is encoded into a small set of values (the latent space), then decoded back to an approximation of the original image as in Fig. 9. One virtual layer of neurons encodes images into actuator settings. The resulting supercontinuum spectra are translated by another layer of virtual neurons into the decoded image. Decoding is a good match for the dimensionality of the system, where a few actuators generate many spectral powers.

Training analog neural networks is slow since the gradients are not easily known [1]. We describe here a method that uses some emulation, and needs limited acquisitions to train the virtual input and output layers. The major steps are:

(1) *Acquire* spectra at random actuator settings;

(2) *Train* a supercontinuum emulator;

(3) *Train* both translators together virtually using the emulator;

(4) *Acquire* training spectra using input translator and training data; and

(5) *Retrain* output translator using training spectra.

First, we acquired 22 660 spectra for random settings of three actuators (two heaters, one pump diode) for training a neural network to emulate the supercontinuum (three inputs, 982 outputs). The emulation only outputs a spectrum, it is not an optical propagation calculation [20]. The spectrometer used here was a broadband Fourier transform spectrometer, and the spectra were averaged and reduced in resolution in software. The emulation works fairly well in overall shape, but not details, as seen in Fig. 10.

Next, the emulator's parameters are frozen, and it replaces the supercontinuum in the autoencoder for a fully virtual neural network. The input and output translation layers are trained with backpropagation in the computer. At this point, replacing the emulation with the actual supercontinuum would autoencode, but likely not well.

Using the standard fashion MNIST image set [21], we use the trained input translator to convert the first 12 881 training images into actuator settings, and measure the corresponding training spectra. With the training spectra as the inputs to the output translator, and the training images as the target, we can train the output translator with backpropagation in the computer. This retraining step ideally corrects for errors introduced by the emulator. Sandwiching the supercontinuum between the original input translator and the retrained output translator completes the hybrid network.

Some autoencoding test samples are shown in Fig. 11, with comparison between the hybrid supercontinuum network (analog); the hybrid network with the supercontinuum removed [Comp. (1), computer only with one hidden layer]; and adding two hidden layers in place of the supercontinuum [Comp. (3), computer only with three hidden layers].

Our goal is not to perform a flawless autoencoding, but to confirm whether the supercontinuum can aid in a neural network calculation. For example, in the common test case of MNIST handwriting decoding [22] (which we have also attempted [23]), the small translation layers already have

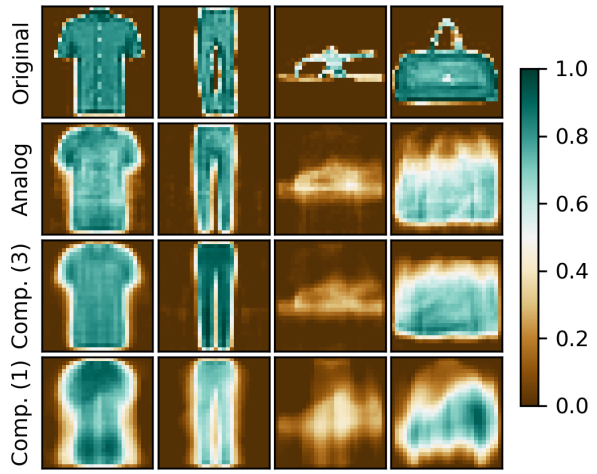

FIG. 11. Originals (top) and images decoded from three values. The second row is the hybrid physical network. The bottom row is the same neuron configuration as the hybrid network, with the supercontinuum removed (one hidden layer). The third row has two layers in place of the supercontinuum (three hidden layers), which produces output closer to the hybrid result.

enough computing power to mostly solve the problem on their own. Here, we compress images down to a latent space of only three values, where autoencoding with the smallest possible translation layers [Comp. (1)] yields poor image quality that leaves room for the supercontinuum to improve performance.

Comparing the analog and Comp. (1) rows shows that the supercontinuum clearly improves the images, with sharper edges and narrower features, such as the shoe straps and the bag handle. Trying a few virtual layers in place of the supercontinuum finds that adding two hidden layers with 32 and 64 neurons makes the computer-only output visibly similar to the hybrid output. This is confirmed numerically in Fig. 12 with the distribution of autoencoding errors for the three cases, with an error of 1 corresponding to the full dynamic range of the image. The analog and Comp. (3) cases both show less error than Comp. (1).

These calculations provide a direct comparison for estimating the virtual neuron equivalence of the supercontinuum computation. The 96 neuron count is much smaller than our crude upper-limit estimate. There are many differences from the delta function experiment, including the specific task, having more layers, and a different spectrometer.

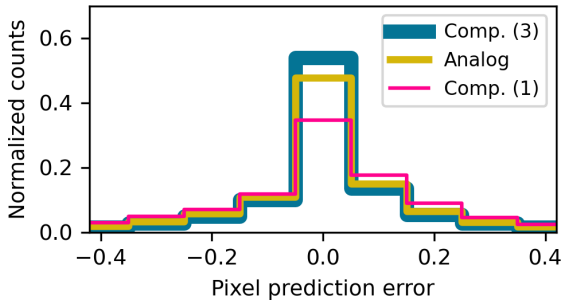

FIG. 12. Histogram of errors (normalized to the dynamic range) for the three autoencoders, quantifying the visual quality visible in Fig. 11, where the supercontinuum improves performance roughly at the level of two virtual neural network layers.

Having autoencoded images with supercontinuum, we can at least say that supercontinuum faithfully transmits information encoded in the inputs. This is important, as it means that our system reliably translates digital data into the spectral domain through a highly nonlinear interaction. High nonlinearity is often avoided in optical computing for perceived environmentally instability, but this was not an issue here. The system had pump diodes with conventional current and temperature stabilization, and a fiber Bragg grating pulse shaper on a plate in a box, both temperature stabilized with standard proportional-integral-derivative (PID) controllers [16]. This measurement included data taken over a month apart.

Given the approximate and averaging nature of neural networks, minor drifts of the physical system or an incoherent supercontinuum causing spectral blurring can be quite tolerable. Taking Fig. 10 as an example, if there are moving spikes within a stable envelope, contributions of the narrow features should average away with sufficient training, as it does with noise in general.

The training method was designed for our case of slow actuation. We wait 20 s when changing the heaters, but then we can acquire at the spectrometer rate of about 1 s when changing only the pump current for that particular heat setting. Speed could be increased to kHz rates by using mechanical force from electrothermal microactuators [24] or piezoelectrics. Supercontinuum is sensitive to almost any type of actuation, so GHz-speed electro-optic modulators could be used in, say, an interferometric design. Fast readout such as detector arrays or temporal reading of a highly chirped pulse would be needed to match.

Our method has the benefit of using only two large datasets, with the training occurring offline rather than open-ended live training. This allows for tweaking of training parameters using the same data, and without discarding data. The need for successful emulation may limit the number of parameters. The emulation only creates the input translator though, so as long as the translator does not severely restrict the input parameter space, it is likely better than manual mapping.

V. CONCLUSIONS

We used an octave spanning supercontinuum as an optical computer with a training method for hybrid analog systems. Our measurements show that supercontinuum can emulate functions, meaning that it can transmit and process data at the level of at least a small neural network, so supercontinuum's processing and wavelength conversion abilities can contribute to optical information systems, particularly if pulse energies and waveguide size can be reduced [25].

Optical computing often focuses on energy efficiency [26], but at this very early demonstration stage where the type of implementation is far from clear, we do not attempt an estimate (Sec. G of the supplement to Ref. [15] has some numbers for a similar system). Supercontinuum may not necessarily be the main inference engine; it may use its processing capabilities to route and modify light between more efficient devices on a chip. Energy is used to make a pulse, shape it, and measure it, but the pulse may already exist, may be preshaped, and would be passed along to the next device rather than be measured. The supercontinuum might then only

contribute waveguide losses, and the actual energy use would be a system-wide property.

Device density is one area where optical computing is constrained by the micron-length scale of the wavelength [26]. The supercontinuum may improve density by performing the equivalent of wavelength multiplexing and multiple interferences in a single waveguide, while device complexity such as addressing would move to the pulse preparation stage. The pulse preparation may occur further upstream, for example, at the transmitter in a routing system. In that case, the receiving supercontinuum would not need any separate information from the transmitter; the instructions and data would both be part of the light packet, which itself controls the supercontinuum process.

Having proven calculation abilities, the unique wavelength conversion and continuous computation properties of supercontinuum greatly expands the optical computing toolbox.

[1] L. G. Wright, T. Onodera, M. M. Stein, T. Wang, D. T. Schachter, Z. Hu, and P. L. McMahon, Deep physical neural networks trained with backpropagation, Nature (London) **601**, 549 (2022).

[2] K. Kitayama, M. Notomi, M. Naruse, K. Inoue, S. Kawakami, and A. Uchida, Novel frontier of photonics for data processing–photonic accelerator, APL Photonics **4**, 090901 (2019).

[3] N. Kirman, M. Kirman, R. K. Dokania, J. F. Martinez, A. B. Apsel, M. A. Watkins, and D. H. Albonesi, Leveraging optical technology in future bus-based chip multiprocessors, in *Proceedings of the 2006 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'06)* (IEEE, New York, 2006), pp. 492–503.

[4] R. Hamerly, L. Bernstein, A. Sludds, M. Soljačić, and D. Englund, Large-scale optical neural networks based on photoelectric multiplication, Phys. Rev. X **9**, 021032 (2019).

[5] L. De Marinis, M. Cococcioni, P. Castoldi, and N. Andriolli, Photonic neural networks: A survey, IEEE Access **7**, 175827 (2019).

[6] G. Wetzstein, A. Ozcan, S. Gigan, S. Fan, D. Englund, M. Soljačić, C. Denz, D. A. Miller, and D. Psaltis, Inference in artificial intelligence with deep optics and photonics, Nature (London) **588**, 39 (2020).

[7] Y. Bai, X. Xu, M. Tan, Y. Sun, Y. Li, J. Wu, R. Morandotti, A. Mitchell, K. Xu, and D. J. Moss, Photonic multiplexing techniques for neuromorphic computing, Nanophotonics **12**, 795 (2023).

[8] C. Huang, V. J. Sorger, M. Miscuglio, M. Al-Qadasi, A. Mukherjee, L. Lampe, M. Nichols, A. N. Tait, T. F. de Lima, B. A. Marquez, J. Wang, L. Chrostowski, M. P. Fok, D. Brunner, S. Fan, S. Shekhar, P. R. Prucnal, and B. J. Shastri, Prospects and applications of photonic neural networks, Adv. Phys.: X **7**, 1981155 (2022).

[9] Q. Xu, T. Mytkowicz, and N. S. Kim, Approximate computing: A survey, IEEE Des. Test **33**, 8 (2016).

[10] J. Liu, Q. Wu, X. Sui, Q. Chen, G. Gu, L. Wang, and S. Li, Research progress in optical neural networks: Theory, applications and developments, PhotoniX **2**, 5 (2021).

[11] G. Marcucci, D. Pierangeli, and C. Conti, Theory of neuromorphic computing by waves: Machine learning by rogue waves, dispersive shocks, and solitons, Phys. Rev. Lett. **125**, 093901 (2020).

[12] I. Oguz, J.-L. Hsieh, N. U. Dinc, U. Teğin, M. Yildirim, C. Gigli, C. Moser, and D. Psaltis, Programming nonlinear propagation for efficient optical learning machines, Adv. Photon. **6**(1), 016002 (2024).

[13] C. Huang, A. Jha, T. F. de Lima, A. N. Tait, B. J. Shastri, and P. R. Prucnal, On-chip programmable nonlinear optical signal processor and its applications, IEEE J. Sel. Top. Quantum Electron. **27**, 1 (2021).

[14] N. A. Silva, T. D. Ferreira, and A. Guerreiro, Reservoir computing with solitons, New J. Phys. **23**, 023013 (2021).

[15] B. Fischer, M. Chemnitz, Y. Zhu, N. Perron, P. Roztocki, B. MacLellan, L. Di Lauro, A. Aadhi, C. Rimoldi, T. H. Falk, and R. Morandotti, Neuromorphic computing via fission-based broadband frequency generation, Adv. Sci. **10**, 2303835 (2023).

[16] K. F. Lee, A. Rolland, P. Li, J. Jiang, and M. E. Fermann, Coherent supercontinuum shaping for multiple wavelength optimization over an octave, Opt. Express **30**, 427 (2022).

[17] W. R. Clements, P. C. Humphreys, B. J. Metcalf, W. S. Kolthammer, and I. A. Walmsley, Optimal design for universal multiport interferometers, Optica **3**, 1460 (2016).

[18] T. Zhou, X. Lin, J. Wu, Y. Chen, H. Xie, Y. Li, J. Fan, H. Wu, L. Fang, and Q. Dai, Large-scale neuromorphic optoelectronic computing with a reconfigurable diffractive processing unit, Nat. Photon. **15**, 367 (2021).

[19] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. **2**, 303 (1989).

[20] L. Salmela, M. Hary, M. Mabed, A. Foi, J. M. Dudley, and G. Genty, Feed-forward neural network as nonlinear dynamics integrator for supercontinuum generation, Opt. Lett. **47**, 802 (2022).

[21] H. Xiao, K. Rasul, and R. Vollgraf, Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, arXiv:1708.07747.

[22] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE **86**, 2278 (1998).

[23] K. F. Lee and M. E. Fermann, Computation using shaped supercontinuum generation within a neural network, in *Nonlinear Frequency Generation and Conversion: Materials and Devices XXII*, edited by P. G. Schunemann, International Society for Optics and Photonics (SPIE, Bellingham, 2023), Vol. 12405, p. 124050H.

[24] K. Udeshi, K.-H. Liao, L. Que, Y. B. Gianchandani, and A. Galvanauskas, Programmable optical wave form shaper on a microchip, Appl. Phys. Lett. **89**, 031120 (2006).

[25] J. Wei, C. Ciret, M. Billet, F. Leo, B. Kuyken, and S.-P. Gorza, Supercontinuum generation assisted by wave trapping in dispersion-managed integrated silicon waveguides, Phys. Rev. Appl. **14**, 054045 (2020).

[26] M. A. Nahmias, T. F. de Lima, A. N. Tait, H.-T. Peng, B. J. Shastri, and P. R. Prucnal, Photonic multiply-accumulate operations for neural networks, IEEE J. Sel. Top. Quantum Electron. **26**, 1 (2020).