Training neural networks with end-to-end optical backpropagation

James Spall

Linacre College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Hilary 2024

Acknowledgements

There are many people I need to thank for helping me get to the conclusion of this thesis. I can't name them all, but if you're in the small handful of people that will actually bother reading it, you're definitely one of them! First of all, this would obviously not have been possible without my supervisor (and spin-out co-founder!) Alex - thankyou for your continuous support over the last 4 years in all aspects of this project. Equally big thankyou to Xianxin, the brains behind the original idea, who has helped every step of the way. From setting up the experiment in an entirely empty lab, to co-founding our spin-out, co-authoring papers, coding, electronics, optical alignment, and even a wedding in the Black Forest! Also to my lab group, especially to Aleksei for keeping me sane during the pandemic lockdowns, Tom for doing so much of the hard graft getting the spin-out up and running, and all the other group members past and present, especially those who would always answer the call of "pub?" - thankyou! Likewise to all my other fellow PhD students, especially members of Linacre and the department who made settling into Oxford four years ago so easy, and who have shared in the highs and lows of PhD life. To Charli, for tolerating me pretending to be a student for another few years after moving in together after lockdown, and to my parents for their endless support in every way (and not just financially!) And finally to Tez, my west highland terrier - without your desperate need for breakfast and a walk every morning there's no chance I would have made it to the lab before lunch each day.

List of Publications

Spall, J., Guo, X., Barrett, T. D., & Lvovsky, A. I., Fully reconfigurable coherent optical vector–matrix multiplication. *Opt. Lett.* **45**, 5752 (2020).

Spall, J., Guo, X. & Lvovsky, A. I., Hybrid training of optical neural networks. *Optica* **9**, 803 (2022).

Spall, J., Guo, X. and Lvovsky, A.I., Training neural networks with end-to-end optical backpropagation. *arXiv preprint* arXiv:2308.05226 (2023).

Abstract

Optical computing is an exciting option for the next generation of machine learning hardware that is fast, parallel and energy efficient. To create a truly all-optical neural network, it is necessary to implement both stages of deployment: inference and training. This in turn requires the ability to construct multiple linear and nonlinear layers, and implement backpropagation - the primary algorithm for training neural networks - in optics. Training with backpropagation requires information to flow forward and backward through the same network, and imposes conflicting requirements on the mathematical function of the activation layers in each direction. Although a straightforward proposition for a digital processor, implementing these functions in optics has remained elusive, and so prevented any demonstration of true end-to-end optical training to date. This thesis builds on a conceptually-simple scheme to overcome this challenge, to show the first practical demonstration of a multi-layer optical neural network that includes end-to-end optical training. Coherent Fourier optics and spatial light modulation is used to implement the linear layers of a neural network, in the form of optical matrix-vector multiplication with real-valued or complex-valued parameters. The phenomenon of saturable absorption is used to perform the nonlinear neuron activations, and backpropagation is performed optically by means of counter-propagating beams of light, which act analogously to the pump and probe beams of doppler-free saturation spectroscopy. The optical network is used to successfully perform a range of standard benchmark classification tasks, after training the network with a variety of schemes that combine the physical system and a digital model in different ways. In doing so the advantages of hardware-in-the-loop training over traditional *in-silico* training are shown; improved network accuracy and resilience to errors. This work helps to confirm the potential of building the next generation of hardware for machine learning with analog optics for both inference and training.

Contents

Appendices

List of Figures

List of Abbreviations

ANN . . . . . Artificial Neural Network

BS . . . . . Beam splitter

CCD . . . . . Charged-Coupled Device

DENN . . . . . Digital Electronic neural network

DMD . . . . . Digital Micro-mirror Device

DOE . . . . . Diffractive Optical Element

ECDL . . . . . External-Cavity Diode Laser

GL . . . . . Grey-level

HWP . . . . . Half wave-plate

IoT . . . . . Internet of Things

LC-SLM . . . Liquid-crystal Spatial Light Modulator

LUT . . . . . Look-up Table

MEMS . . . . Micro-Electromechanical System

MNIST . . . . Modified National Institute of Standards and Technology

MVM . . . . . Matrix-Vector Multiplication

ONN . . . . . Optical neural network

p-VVM . . . . Parallel Vector-Vector Multiplication

PBS . . . . . Polarising beam splitter

QWP . . . . . Quarter wave-plate

ReLU . . . . . Rectified Linear Unit

RMSE . . . . . Root Mean Square Error

SLM . . . . . Spatial Light Modulator

SNR . . . . . Signal-to-Noise ratio

TA . . . . . Tapered Amplifier

1 Introduction

Contents

1.1 AI, machine learning and neural networks

Given the incredible rate at which it has become involved in so many aspects of modern society, it seems almost unnecessary to introduce the concept of AI. Over just the past four years during the course of my DPhil studies, we have seen a huge number of transformative announcements in a wide variety of fields: the demonstration of AlphaFold predicting protein structures with remarkable accuracy [1]; the rapid development of generative AI, with the widely-publicised release of ChatGPT, and the astonishing ability of large language models like

GPT-4 to produce images and text almost indistinguishable from human-created content [2, 3]; the ability to discover new drugs, materials and algorithms, with AI models developing new antibiotics to kill deadly 'superbugs' [4], designing new nanostructures through molecular self-assembly [5], and creating fundamentally better ways to sort numbers [6] and perform matrix arithmetic [7].

Many of these recent breakthroughs rely on artificial neural networks (ANN), a class of machine learning algorithm that has now become synonymous with the term AI. ANNs excel at problems such as pattern recognition, classification, regression, and decision-making. Their strength lies in the ability to learn the complex relationships hidden within raw input data, and extract high-level features and characteristics that aren't otherwise obvious to a pre-programmed algorithm [8].

Investigations of neural networks began as early as the 1940s, originally attempting to mimic the behaviour of the brain, with the structure and function of biological neurons inspiring the concept [9–11]. The study and development of modern neural networks began in the early 2000s [12], and we now consider the most general ANN to consist of multiple layers of artificial 'neurons' interconnected by tunable parameters called weights. An ANN takes input data, such as an image, a block of text or the position of pieces in a game, and tries to correctly output a prediction: the object in the image, the next word in the sentence, or the optimal next move in the game. The network starts with random weights and inevitably gives bad predictions, but the power of neural networks is in the ability to learn the optimal weights that give accurate predictions. This process of learning the correct weights for a given problem is called network training.

There are many forms of ANN training, including reinforcement learning, unsupervised learning, and supervised learning, and the choice of training method is dependent on the model and problem being solved. Throughout this work we focus on supervised learning, a very common form of training that is well suited to solving classification and regression tasks. The network is provided with a large dataset of examples where the correct answer is already known, and each example is used to change the weights to improve the network prediction. The most common

method of achieving this is using gradient descent to minimise a loss function - a 'landscape' function that encodes the difference between the network output prediction and the actual ground truth answer. Doing so requires calculating the gradient of the loss function with respect to all the network weights, which is usually achieved with the backpropagation algorithm [13], a simple but extremely important algorithm responsible for the incredible ability of neural networks to learn difficult problems so efficiently [8].

The primary difficulty in many real-world applications of ANNs that use supervised learning is not in designing or implementing the neural network itself, but in collecting or generating a large enough dataset of 'good' data with known answers [14]. The exact volume of data required is dependent on the task being solved, and can range by many orders of magnitude. For example, in the extreme case some datasets used to train image classification ANNs can contain hundreds of millions of images annotated with billions of labels [15]. In comparison, the image classification tasks performed in this work are far simpler, and can be trained using less than $100,000$ images.

The ultimate goal of training a neural network is for the network to accurately replicate the mapping between the network inputs and the ground truth answers, such that when applied to new unseen data, the network output still matches the (unknown) ground truth. This process of using the trained neural network to make predictions on new data is referred to as network *inference*.

The exact structure of neurons and interconnections in a network is referred to as the network architecture. The most general architecture, where layers are joined sequentially and all the neurons in each layer are interconnected, is referred to as a fully-connected feed-forward network, or multi-layer perceptron (MLP), and an example is shown in Fig. 1.1. Networks with more specifics architectures have been shown to improve performance in certain tasks. For example, convolutional neural networks (CNNs) perform particularly well in image classification tasks [16], recurrent neural networks (RNNs) perform well on time-series data [17], and the transformer architecture has been crucial to the performance of large-language

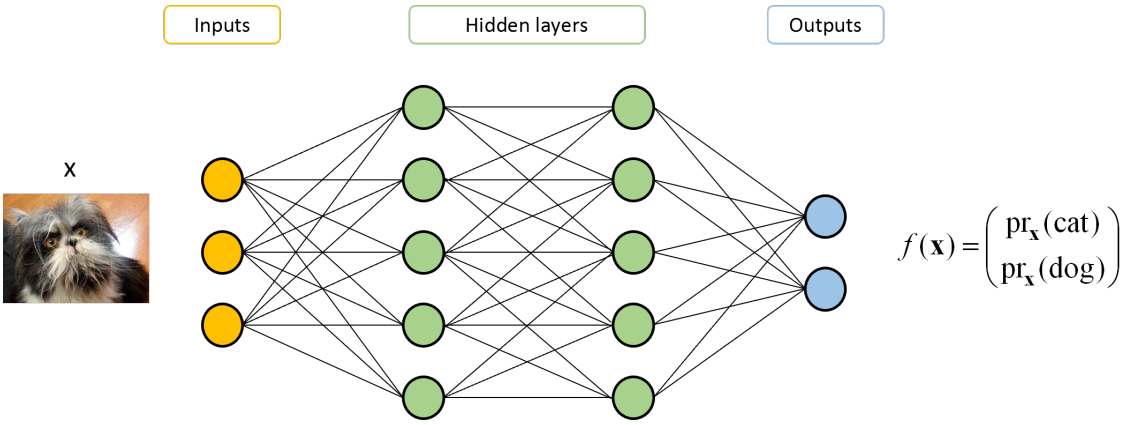


Figure 1.1: A simple feed-forward artificial neural network, or multi-layer perceptron. The network represents some mapping $x \rightarrow f(x)$ designed to perform a certain task; in this example classifying an image as a cat or a dog. The network is formed from layers of neurons, interconnected with tunable parameters.

models such as GPT-4 [2, 18]. Additionally, the architecture of an ANN is often closely associated to the method used for training, as both the architecture and training method are optimized for different types of task. For example, since CNNs perform well at image classification, they are usually trained via supervised learning. However it is important to note that the network architecture is fundamentally independent of the training method used, and many architectures can be utilised with both supervised and unsupervised learning; as an example LLMs often require a combination of both unsupervised learning for feature and pattern analysis within text, and supervised learning for fine-tuning the model output.

Common to all ANN architectures is the structure of alternating linear and nonlinear layers. The weighted interconnection of neurons is a linear process and is represented by a matrix-vector multiplication (MVM). The input to each layer is a vector, with each vector element representing one neuron. The network weights form a *weight matrix*. Each element of the resulting output vector from multiplying the weight matrix and input vector, represents a neuron in the next layer of the network. Building up many such layers forms a 'deep' neural network. Crucially, after each layer, a nonlinear *activation* function is applied individually to each neuron. Without some such nonlinear function, the network represents only a

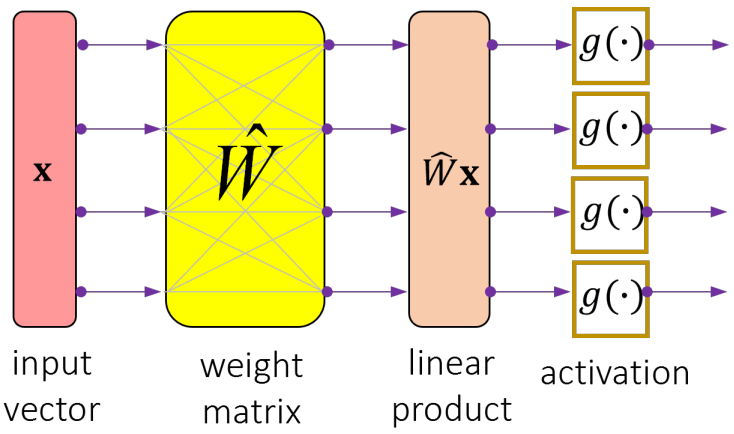


Figure 1.2: The structure of one layer in a feed-forward neural network. The input to each layer x is multiplied by the weight matrix W, encoding the tunable network parameters. A nonlinear activation function is element-wise applied to the resulting linear product.

linearly mapping of inputs to outputs, and is therefore extremely limited. The structure of each layer is shown in Fig. 1.2.

Although conceptually straightforward, neural networks are extremely computationally demanding, with the largest models today created with hundreds of billions of individually tunable parameters, requiring computers that perform trillions of operations per second [19]. The current scale and performance of neural networks can be directly attributed to the increase in available computing speeds over time, which has been enabled by several factors. Most importantly is the rapid increase in microchip transistor density, commonly quantified by Moore's Law: the observation that the number of transistors on a chip doubles approximately every two years [20]. This has held true for many decades, constantly increasing the fundamental speed of digital electronic compute hardware [21].

In addition, the adoption of bespoke hardware for parallel processing has rapidly accelerated the scale and performance of neural networks. In 2012, the winner of a global competition of machine learning algorithms to classify the large ImageNet dataset was won by an ANN architecture named AlexNet, and its success was attributed to the use of graphics processing units (GPUs) to efficiently train the network [22]. Originally designed for rendering graphics, GPUs perform simple arithmetic in a highly parallel fashion, providing the ability to accelerate ANN

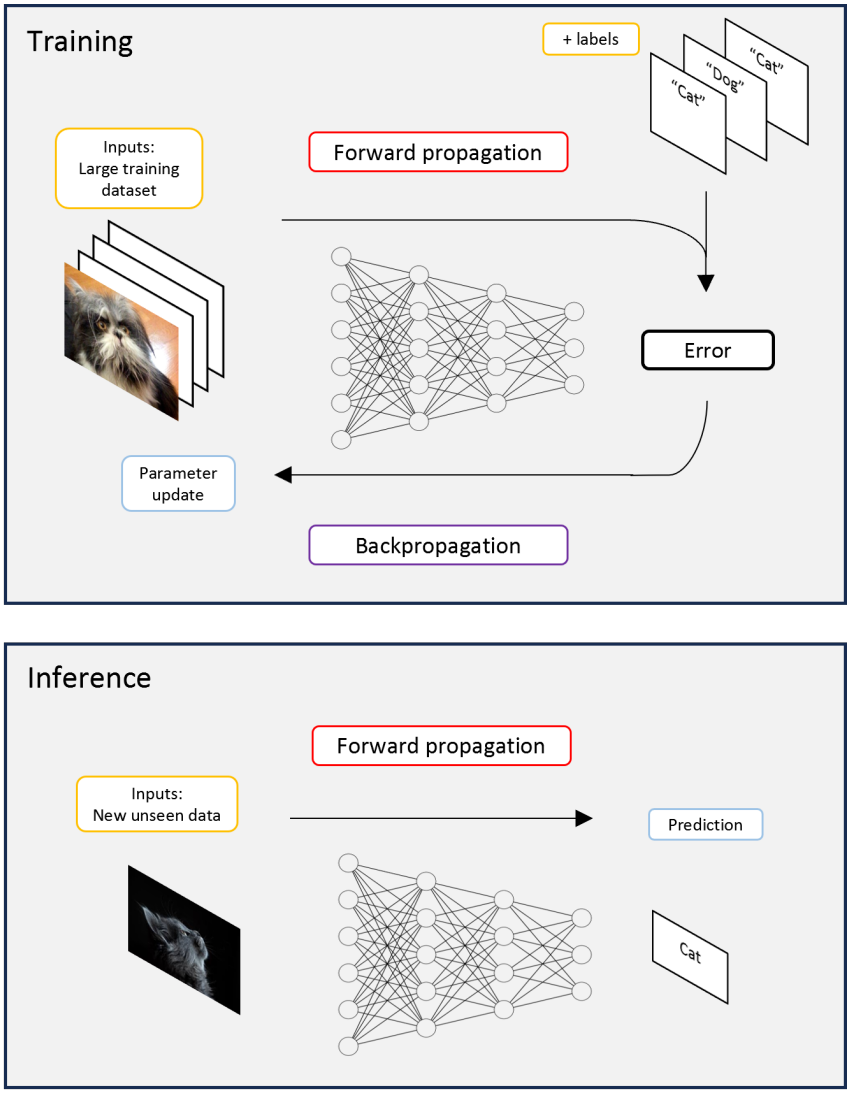


Figure 1.3: Neural networks are deployed in two stages: training and inference. Training, in the most common form of supervised learning, requires a large set of data with known outputs (labels), and consists of forward and backpropagation through the network. Inference is performed on new unseen data and requires only forward propagation.

training, and the breakthrough of AlexNet led to the adoption of GPUs as the default hardware for network training [23]. Even more bespoke and specialized hardware for ANNs has since been developed, including field-programmable gate arrays (FPGAs) [24] and application-specific integrated circuits (ASICs) [25], an example being Google's tensor processing unit (TPU) [26]. All such bespoke hardware is developed with a view to further increasing compute speed and energy efficiency.

Despite the continued advancements over many decades, the ever-increasing

trend of faster computing speeds, and in turn bigger and better AI models, is not guaranteed. In fact the rate of improvement has already begun to slow considerably, as digital electronic hardware reaches its physical limits [27–29]. As the size of each transistor now reaches the scale of just a few atoms, reliability issues, from effects including quantum tunnelling, will prevent transistor density from increasing [30], and the clock speeds of digital electronics has not increased significantly in the past 10 years.

The shift to parallel processing architectures has been critical to increasing compute speeds. However optimising the interconnection between processor cores, and between processors and memory, is practically challenging [31]. More fundamentally, there is a limit to the achievable speed-up from splitting a computational problem across parallel processors if some fraction of the problem cannot be parallelised, however small that fraction might be, as described by Amdahl's law [32, 33]. Therefore utilising parallel processors for ANNs that contain non-parallelizable operations, such as recurrent layers in RNNs, or sequential attention mechanisms in transformers, has fundamentally limited benefit.

Another important consideration is the energy required to power such vast computing systems. The fastest processors now require kilowatts of power each, and in total the energy used in training and running AI models is now a significant fraction of the planet's total [34]. One recent analysis estimated that training a large language model (LLM) with 176 billion parameters released 50.5 tonnes of CO_2 [35]. Whilst some ANNs may be trained only periodically before being used for inference, it is common for networks to undergo continuous retraining as new data becomes available, and the inference stage can equally require huge computational power. The growth of ANNs and the computing resources required to power them has obvious consequences for sustainability, with growing concern surrounding the carbon footprint associated with training and deploying AI models [36].

Modern-day neural networks form a massive and complicated field of study, including the research of new algorithms and architectures, data science techniques, application areas, and novel forms of computing hardware. Given the fundamental

issues faced by digital electronic hardware, the study of analogue processors has seen a great deal of renewed interest, especially for the linear arithmetic operations essential for neural networks, and particularly in the domain of optics rather than electronics, where light is the underlying carrier of information instead of electrons [37]. The idea of optical computing is not new, nor is its application to machine learning, but with the rapid rate of development of AI, an ever-growing need for fast and energy-efficient processors, and the availability of sophisticated modern-day opto-electronic technologies, optical computing is an extremely compelling option for the next generation of AI hardware, with huge potential benefit in computing speeds and energy efficiency [38–40].

1.2 Optical neural networks

1.2.1 History and overview

There is already precedent for using light instead of electronics to carry information, in the field of communications. Optical fibres have been revolutionary for high-speed, long-distance data transmission, with optics providing significantly higher bandwidth, lower signal attenuation and faster transfer rates than copper cables. Beyond traditional communications, they are now used extensively in data centres, with their ability for high-speed, efficient and reliable data transmission making them ideal to interconnect servers, switches, routers and storage systems within giant supercomputing clusters [41].

Going a step further, and using light to not only transfer information, but also *process* information, is a natural progression that has been continuously explored for over 60 years [42]. Digital optical computing architectures have been explored [43, 44], but creating transistor-like components with optics is extremely challenging, especially in an efficient and scalable manner [45]. Instead, it was recognised that using optics in an analog fashion was far more promising, and the 1980s and 90s - the ‘golden age’ of optical computing - saw the development of many novel implementations [38, 42]. In particular, optics was demonstrated to be an ideal platform for performing linear operations, such as Fourier transforms [46],

convolutions [47], and structured arithmetic, in the form of matrix-vector and matrix-matrix multiplication [48, 49]. These types of functions form the backbone of ANNs, and there was therefore great interest in applying optical computing techniques to create physical analog neural networks using light: optical neural networks (ONNs) [48, 50–53].

After a decline in interest as digital computing systems became ever-more powerful, the study of ONNs has become an extremely active area of research once again. In the last few years, there have been many demonstrations of ONNs, using different opto-electronic devices, the different properties of light, and implementing different network architectures. Many have implemented 'standard' network architectures, including fully-connected [54–61], convolutional [62–65] and recurrent [66] ONNs. Our experiments and this thesis focus on feed-forward neural networks, i.e. implementing fully-connected ANN architectures with optics. We detail some of the important realisations of such ONNs in the next section.

In contrast, many groups are combining optics and electronics in new architectures such as reservoir computing [67] and spiking neural networks [68] that have no current parallel in digital hardware. These *neuromorphic* processors and architectures are heavily inspired by biology and the processes that occur in the brain, in a similar fashion as were originally digital ANNs, and are promising avenues to overcome fundamental constraints of digital hardware, such as the Von Neumann bottleneck [69].

The development of optical computing and ONNs is tightly connected to the development of key enabling technologies. The invention and development of the laser and subsequently computer-generated holography was critical to the early growth of optical computing, providing coherent light sources that can be easily manipulated [70–72]. Later came rapid improvement in spatial light modulator (SLM) technology, including faster refresh rates, higher pixel resolutions, and greater modulation bit-depths [73–75]. This enabled more precise and larger-scale encoding of information into the optical domain, enabling many of the breakthroughs in free-space optical computing. Finally, the relatively recent development of integrated

photonics has greatly expanded the field of ONNs. Instead of using traditional optical elements such as lenses and mirrors in combination with SLMs to modulate a free-space beam, integrated photonics uses chip-scale components to modulate light coupled into waveguides [76–78]. This brings numerous potential benefits, including the ease and speed with which network parameters can be configured and controlled, and the potential to closely integrate optical and traditional electronic computing systems [79].

Regardless of the architecture or implementation, all ONNs use either the interference properties or diffractive properties of optics to connect neurons in a network. Performing this interconnection in an analog fashion using optics has the potential for enormous improvements in speed and energy efficiency [38–40]. Convolutional ONNs are already matching digital networks in terms of speed, with one demonstration reaching over ten trillion operations per second (TOPS) [64]. The energy consumption from optical processing is extremely low, with one recent demonstration showing it is possible to perform MVM with such low optical power that the average photon number per multiplication is less than one, when averaged over all spatial modes [60]. A recent analysis on the use of optics in transformer neural networks suggests that using optics could increase the energy efficiency by two orders of magnitude over current networks, and over 8000 times on larger future networks [80].

However there are still very few, if any, practical implementations of ONNs that outperform digital networks on real-world problems, and creating optical, photonic or analog devices at scale is challenging. In contrast to digital systems, issues such as noise and device variability must be considered when scaling up analog systems [81, 82]. Additionally, the conversion of data between the digital and analog domains requires significant energy, such that the efficiency benefits of performing the calculations optically may be lost. The number of digital-analog conversions must be minimised, which suggests implementing the whole network, including the nonlinear functions, should be performed optically. This is challenging to realise experimentally, due to the limited and complex nature of nonlinear optical

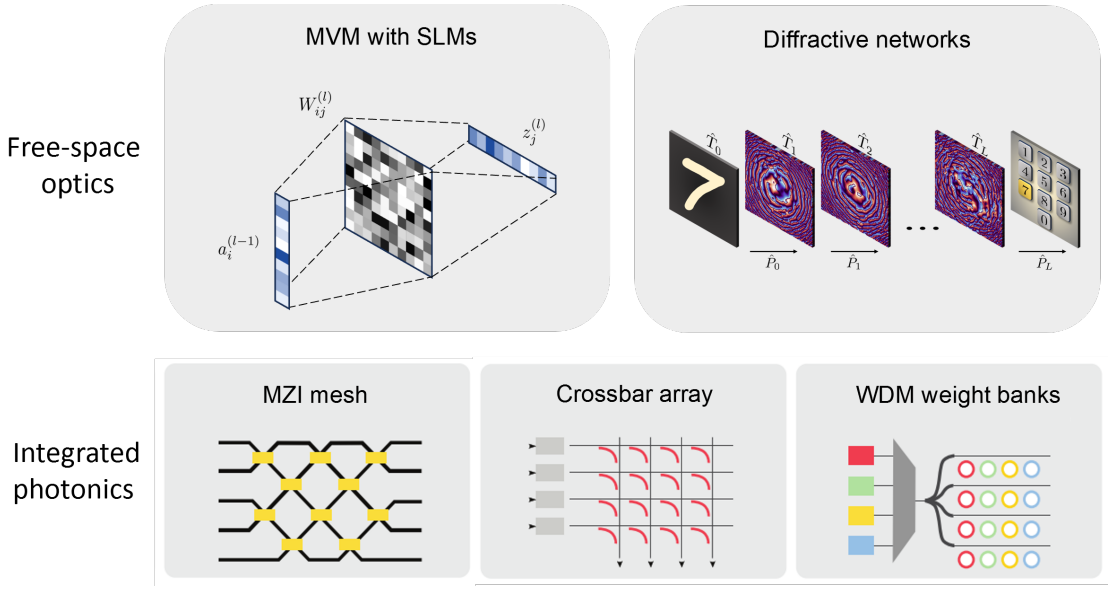


Figure 1.4: Conceptual schemes for implementing matrix multiplication in various optical and photonic platforms. Free-space optics and spatial light modulators can be used to directly implement MVM for feedforward networks by modulating beam transmission, or used to control beam diffraction in diffractive networks. Integrated photonics can be used to construct optical networks with light propagating through waveguides, utilising the interference and coupling of light between different channels to implement MVM.

phenomena. However, a few such nonlinear effects have been demonstrated in ONNs, including electromagnetic induced transparency (EIT) [56], intensity detection of complex-valued fields [57, 59], saturable absorption and gain in different media [50, 83, 84], and reconfigurable opto-electronic nonlinearities [85–87].

Finally, the majority of ONNs demonstrated to date are only used to perform network inference [88]. The training of ONNs, and the use of analog hardware in general for training neural networks, is an important field of study which has already received a great deal of attention but with many outstanding challenges. In section 1.3 we discuss the training of ANNs and ONNs in greater detail.

1.2.2 Example implementations of feed-forward ONNs

Fully-connected feed-forward neural networks are the most general form of ANN. Information flows sequentially through layers of neurons, with all-to-all neuron connections between the layers. These networks are extremely powerful and ubiquitous across different models, and it has been proven that given a large

enough and deep enough fully-connected network, any continuous function can be realised. This is referred to as the universal approximation theorem [89]. Each layer of a feed-forward network takes the form a matrix-vector multiplication (MVM). The N input neurons form a vector $\mathbf{v}$ of size $N \times 1$, the M output neurons a vector $\mathbf{u}$ of size $1 \times M$, and the network weights interconnecting the neurons form an $N \times M$ matrix $\mathbf{W}$. The required calculation is then $\mathbf{u} = \mathbf{W}\mathbf{v}$.

There have been many demonstrations of ONNs with fully-connected architectures, with different implementations promising various benefits. In each case the linear interconnection of neurons takes the form of optical MVM, performed by manipulation of an optical signal as it propagates through the ONN. Different implementations utilise different optical properties: transmission and attenuation, interference, coupling strength, or diffraction [90]. Constructing an ONN then requires multiple such MVMs to be cascaded, and some form of optical nonlinearity introduced. Many excellent reviews can be found on these implementations and ONNs in general [37, 88, 90–93], and some of the implementations are summarised below.

Free-space optics with SLMs (transmission-based MVM)

Schemes to achieve MVM using free-space optics and SLMs were first conceived in the 1970s, and the most common is often referred to as the 'Stanford Multiplier' following Goodman's design [46]. Vector and matrix elements are spatially encoded in the amplitude or intensity of a beam, which can be coherent or incoherent, using an SLM. This can be as simple as a transparency mask, or modern devices such as liquid-crystal displays and micro-electromechanical system (MEMS) devices. Multiplication is achieved as the beam passes through (or reflects from) multiple SLMs, determined by the transmission or attenuation of the beam by the SLM. The scheme also relies on optical fan-out and optical fan-in, the process of spreading and converging beams with lenses. Many demonstrations have shown variations of this, using microlens arrays or cylindrical lenses to perform fan-out and fan-in [56, 60, 63, 66, 84, 94, 95].

SLM-based MVM is the most direct analogy of digitally-implemented MVM, with every vector and matrix element mapped one-to-one to an addressable opto-electronic device, for example by associating each SLM pixel value to one matrix element. SLM-based MVM gives the best potential for maximising the number of neurons implemented in each layer, which is ultimately limited by the achievable pixel resolution of the SLM. Resolutions of modern dynamically-reconfigurable devices are typically on the order of millions of pixels. By way of example, one recent demonstration performed an optical vector-vector dot product where an SLM was used to encode a vector with dimension over 0.5 million [60], although it is important to note this was not a true implementation of free-space MVM with optical fan-out and fan-in.

We implement a variant of free-space optical MVM in our experiments [96], and extensive details on the concept and methods are presented in Chapter 2. Our experiments achieve some of the largest MVM matrix dimensions demonstrated, up to 200×50, use coherent light such that real-valued and even complex-valued elements can be encoded, and maintain excellent precision whilst being arbitrarily reconfigurable.

SLM-based MVM can been used to construct multi-layer ONNs. A few such ONNs have been shown in practice, although most demonstrate only a two-layer network. A variety of nonlinear effects can be used as the activation in such ONNs. Zou et al. used EIT [56] as the nonlinearity in a two layer network [56, 95], and Wang et al. used saturable gain nonlinearity from an image intensifier [84]. In our work, we use saturable absorption in a rubidium vapor cell as the activation function in a two-layer ONN.

MZI mesh (interference-based MVM)

Using integrated photonics requires a different approach to performing MVM, and many demonstrations rely on the concept of cascaded Mach-Zender interferometers (MZIs) [54, 59, 61, 97, 98]. In this scheme, each element of the input vector is represented by attenuating a beam coupled into a waveguide. The vector can then

be multiplied by a unitary matrix, realised by a combination of many on-chip MZIs, each formed from phase-shifters and beam splitters, arranged in a mesh that couples the waveguides and interferes the beams. Arbitrary matrices can be realised by decomposition into unitary and diagonal matrices using singular-valued decomposition (SVD). Unlike the SLM-based free-space multiplier, there is no one-to-one correspondence between a particular optical element and a neuron or network weight. Instead the entire MZI mesh acts to implement the MVM. Previous work has explored different mesh designs [76, 77], susceptibility to fabrication errors and fault tolerance [81, 82, 99], implementation of complex-valued weights [59], and application to a range of neural network tasks [54, 61].

Using integrated photonics gives access to much faster modulation speeds compared to SLM-based designs. SLMs can usually operate no faster than kHz speeds, with liquid crystal devices typically even slower with video refresh rates around 60Hz. In comparison, integrated photonic modulators have been shown to operate at 10s of GHz speed [100–103]. However a drawback of these MZI mesh designs is that the number of components through which light must travel grows quadratically with the size of matrix being encoded [77]. This raises potential issues with noise accumulation and fabrication defects limiting the scalability of such systems [82]. Whereas SLM-based ONNs can encode thousands of neurons per layer, MZI mesh ONNs are limited to the order of tens of neurons per layer [54, 92].

MZI mesh MVMs can be cascaded to form multi-layer ONNs, without the need for optical-electronic conversion or coupling the light on and off the chip. However given the challenge of fabricating a large number of MZIs on a single chip, there are few demonstrations of this in practice. Shen et al. demonstrated a two-layer on-chip ONN [54], although the nonlinear activation was simulated digitally. Recently a fully-optical three-layer ONN using a novel opto-electronic nonlinearity between the layers, all built on a single chip, was demonstrated [61].

WDM weight banks and crossbar arrays (coupling-based MVM)

MZI meshes are not the only way to implement MVM with integrated photonics. A key advantage of using optics over electronics is the ability to use wavelength-division multiplexing (WDM) - encoding information onto different wavelengths of light. This is critical in many integrated photonic designs, which use microring resonators (MRRs) acting as tunable filters to multiply different vector elements by different weight values [64, 68, 104–109]. A common scheme is to implement a crossbar array of modulators, similar to the systolic array design of some digital MVM hardware [110]. Other methods of selective coupling between waveguides without using MMRs have been demonstrated, such as the use of phase-change materials, opening the potential to perform in-memory computing [65, 111].

This was the scheme used to implement an 11 TOPS optical convolutional processor [64], utilising the ability to combine WDM for parallel processing with extremely high modulator speeds to achieve huge throughput, although with a convolutional, not fully-connected, architecture. However the physical footprint of the MMR components creates similar concerns for scalability as with MZI mesh designs, plus the fabrication process may be more difficult at scale as each MMR requires a different and finely-tuned splitting ratio.

Finally, the construction of a multilayer ONN using this design is possible, but requires some form of optical-electronic-optical conversion (though still remaining an analog process), as the different wavelength signals are accumulated by a photodiode to complete the MVM. A nonlinear function can be applied with analog electronics before the electrical signal is used to modulate the input to the next ONN layer. Such schemes have been proposed [107] and successfully demonstrated [109].

Diffractive networks (diffraction-based MVM)

Finally, we describe a slightly different implementation of a feed-forward ONN, which unlike all the methods described so far, does not directly implement the neuron interconnection as an MVM. Deep diffractive neural networks (D^2NNs) [55] use many layers of diffractive optical elements (DOEs) to sequentially modulate a

beam as it propagates and diffracts [55, 57, 58, 112–119]. The entire system acts to implement a linear transformation, which is determined by tuning the amplitude or phase of each DOE through some iterative training process. This requires precise modelling of how the beam propagates through the system [112]. Each diffractive layer cannot perform arbitrary MVM individually, however with a large number of diffractive layers acting sequentially, an arbitrary linear transform equivalent to an MVM can be realised [113]. Similar terminology with different meaning is used in these diffractive systems and other ONNs, which can cause confusion: many *diffractive layers* must be used in unison to create a single *neural network layer*, and similarly each diffractive layer contains many *diffractive neurons* which do not correspond directly to *network neurons*.

D^2NNs are an attractive option for real-time image processing, as they can be used to directly process light from objects, without complex encoding methods or coupling schemes [120]. Additionally, the full resolution of DOEs or SLMs can be used, and billions of diffractive neurons can be interconnected, allowing the networks to be extremely expressive [118]. However training such systems requires true modelling of the diffraction of the beam, which is incredibly computationally expensive, and most demonstrations to date are limited to only a few diffractive layers.

1.2.3 Optical advantage and potential for scaling

Some technologies have a clear definition of what underpins their potential advantage in comparison to traditional digital processors based on silicon chips. For example, quantum computing leverages the inherently different underlying principles of quantum mechanics to perform new algorithms that are simply unachievable with classical digital processors. However there is still a degree of confusion in the field of optical computing as to where the fundamental advantage over digital chips actually derives.

It is also important to note that the field of AI today consists of a huge variety of algorithms and neural network architectures, making a comparison between the speed and energy efficiency of different types of hardware extremely difficult, even

between traditional digital electronic devices. Furthermore, the largest models today require trillions of arithmetic operations, and it is impractical to train or deploy such a model with a single processor. Even when considering future optical or analog devices, one should expect the largest models to require a similar system of many interconnected analog processors, with the model decomposed to run across the devices in parallel.

However we can still consider how analog devices may scale with the size of a problem, in order to identify the root of any potential advantage. In particular, in order to outperform digital processors the scaling between number of mathematical operations performed per physical action must be improved. In this context physical action refers to operating a transistor, MZI or laser modulator, for example. An important recent work by McMahon [121] lists 11 features of optics and optical computing systems that may contribute to outperforming digital processors, particularly in the field of AI and machine learning where analog processors are well suited. Of particular relevance is the argument that optical advantage may result from the combination of improved spatial parallelism and the process of optical fan-out within optical matrix-vector multiplication.

Free-space optics in particular is able to benefit from the utilisation of all three spatial dimensions. Details of the principal of free-space optical MVM and optical fan-out are given in the next chapter, but the crucial idea is that information is encoded in a plane tangential to the beam propagation. One can argue this provides a quadratic scaling advantage in the number of mathematical operations performed per physical action, since the number of multiplications performed is proportional to the square of the number of vector elements. Consider an MVM with a fixed square matrix of dimension $N \times N$. Then N physical actions are required to encode the input vector, for example encoding each element by modulating the intensity of an optical beam. However optical fan-out enables N^2 multiplications to be performed. The challenge therefore is scaling N to such an extent that the ratio between number of arithmetic operations performed, and number of physical actions, is large enough to outperform digital electronics. McMahon estimates this requires a matrix dimension

larger than $10^4 \times 10^4$ [121]. The current bottleneck to achieving such a scale is the pixel resolution of SLMs, which typically have dimension of order $10^3 \times 10^3$.

Optical fan-out therefore provides a quadratic scaling advantage by allowing many multiplications to be performed per physical encoding. However it has been shown that performing optical fan-in may limit such advantage. Each row of the $N \times N$ matrix must be converged to a detector, and the conservation of Étendue, a fundamental optical principle, dictates that only a fraction of power proportional to $1/N$ can be collected by the detector. This assumes the beam is converged to an equivalent cross sectional area as the input, and can therefore be mitigated depending on the exact implementation of optical fan-out and fan-in. However when cascading multiple MVMs, this means some optical loss is fundamentally unavoidable, even in an ideal experiment, which in turn may offset the advantage offered by optical fan-out. First noted by Goodman in the 1980s [122], this is a subtle but important problem to consider for the scalability of ONNs built from cascaded MVMs.

Similar considerations must be made when considering the scalability of other optical implementations. For example it appears that using an integrated photonics MZI mesh for MVM does not suffer from the same power loss issue from optical fan-in. However this is only true in the case of implementing unitary matrices; using SVD to implement an arbitrary matrix requires attenuation of the optical signals for the diagonal matrix, in which case energy loss is unavoidable.

In the near term, it appears the main consideration for the scalability of different technologies will involve practical or material engineering challenges. For example the current pixel resolution of SLMs may be an order of magnitude smaller than needed for optical advantage, and the number of integrated photonic components that can be fabricated on a silicon wafer several orders of magnitude smaller. Other technology-dependent overheads, such as the energy consumption in converting between analog and digital signals, must also be considered when making comparisons. Understanding how optical and other analog devices may outperform their digital electronic counterparts is not easy, and warrants continued investigation alongside and in combination with the development of the technology itself.

1.3 ONN training

1.3.1 Training ANNs: Gradient descent and the backpropagation algorithm

The ability of ANNs to learn their optimal parameters through training is what distinguishes them from traditional problem-solving algorithms. Throughout this work, we take training to mean the process of supervised learning, whereby the parameters of a neural network are iteratively updated via gradient descent using a labelled dataset of inputs and associated ground truth answers. Here we outline the process of training a simple feed-forward ANN using the backpropagation algorithm [13], which is essentially an application of the chain rule of calculus, and allows the network to find the most 'efficient' way to update its parameters to minimise a chosen loss function.

Supervised learning requires a labelled dataset $(\mathbf{x}, \mathbf{t})$, where $\mathbf{x}$ are example network inputs, and $\mathbf{t}$ is the associated known label, giving the ground truth answer. First, an example from the training dataset is sent to the network input, $a_i^{(0)} = x_i$. The neuron values at all subsequent layers are then interconnected by weight matrices $W_{ji}^{(l)}$ as

$$z_j^{(l)} = \sum_i W_{ji}^{(l)} a_i^{(l-1)}. \tag{1.1}$$

A nonlinear *activation* function $g(\cdot)$ is applied element-wise to each neuron,

$$a_j^{(l)} = g(z_j^{(l)}). \tag{1.2}$$

We see that each layer of a neural network is therefore simply an MVM, where the input vector is the previous layer activation, and the matrix elements are the network weights in that layer. The activation function can take any nonlinear form, but some common functions are listed in Tab 1.1. Different functions can be used at different layers.

For each input example, we measure and store the activations $a_j^{(l)}$ at every layer, including the network output $y_j = a_j^{(L)}$ for a network with L layers. A loss function, $\mathcal{L}(\mathbf{y}, \mathbf{t})$, is defined in order to quantify the divergence between the network

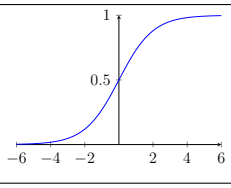
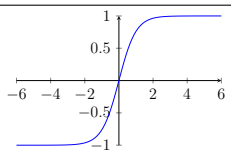
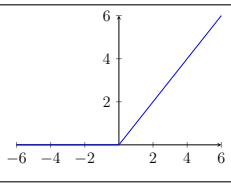
Name	Function	Graph
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	
Hyperbolic Tangent	$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	
Rectified Linear Unit (ReLU)	$f(x) = \max(0, x)$	

Table 1.1: List of common nonlinear activation functions used in deep neural networks.

output and the associated label. The exact form of the loss function can vary, and is often tailored to the particular problem the network is trying to solve. For example, classification tasks often utilise the 'softmax' activation function at the final layer, and categorical cross-entropy (CCE) as the loss function. For regression problems, its common to use no output activation at all, and a typical loss function is mean-squared error (MSE). Further details are provided in Appendix A.

The next step in network training is to use the backpropagation algorithm to calculate the gradient of this loss function with respect to all the network weights. The gradients we require are given by

$$\frac{\partial \mathcal{L}}{\partial W_{ji}^{(l)}} = \sum_k \frac{\partial \mathcal{L}}{\partial z_k^{(l)}} \frac{\partial z_k^{(l)}}{\partial W_{ji}^{(l)}} \tag{1.3}$$

$$= \delta_j^{(l)} a_i^{(l-1)}, \tag{1.4}$$

where $\delta_j^{(l)} \equiv \partial \mathcal{L}/\partial z_j^{(l)}$ is referred to as the 'error' at the j-th neuron in the l-th layer. Equation (1.4) is very important; it tells us that to calculate the weight updates at each layer we need one vector from the 'forward pass' through the network (the activations) and one vector from the 'backward pass' (the errors).

To find the latter, we first calculate the error at the final layer directly from the loss function and network output:

$$\delta_j^{(L)} = \frac{\partial \mathcal{L}}{\partial z_j^{(L)}} = \frac{\partial \mathcal{L}}{\partial y_k} \frac{\partial y_k}{\partial z_j^{(L)}}. \tag{1.5}$$

In the case of using softmax activation function and CCE loss function this simplifies nicely to

$$\delta_j^{(L)} = y_j - t_j. \tag{1.6}$$

A detailed derivation is provided in Appendix A.

We now use this output error to calculate the errors at all previous layers, moving sequentially backward through the network, by applying the chain rule of calculus. We have

$$\delta_j^{(l)} = \frac{\partial \mathcal{L}}{\partial z_j^{(l)}} \tag{1.7}$$

$$= \sum_{k,m} \frac{\partial \mathcal{L}}{\partial z_k^{(l+1)}} \frac{\partial z_k^{(l+1)}}{\partial a_m^{(l)}} \frac{\partial a_m^{(l)}}{\partial z_j^{(l)}} \tag{1.8}$$

$$= \left(\sum_k \delta_k^{(l+1)} W_{kj}^{(l+1)} \right) g'(z_j^{(l)}). \tag{1.9}$$

We find that the error at layer l is connected to the error at layer $l+1$ by an MVM, mirroring exactly the same operation performed in the forward direction. However instead of applying the activation function, we element-wise multiply by the gradient of the function: $g'(z_j^{(l)})$. Given the output layer error, we can now find the errors at all previous layers.

The activations and errors calculated from the forward and backward passes are combined to calculate the gradients (1.4). These inform us how to update the weights in order to minimise the loss function:

$$W_{ji}^{(l)} \leftarrow W_{ji}^{(l)} - \eta \left(\frac{\partial \mathcal{L}}{\partial W_{ji}^{(l)}} \right), \tag{1.10}$$

where η is the *learning rate*. The weights can be iteratively updated via gradient descent by repeating this procedure for every labelled example, until the loss function

reaches a minimum value. At this point the network has converged and encodes the optimal parameters. In practice, multiple inputs are run through the network in *mini-batches*, and a single weight update is calculated using the average of each mini-batch. The entire training dataset may be passed through the network multiple times. Each pass of the entire dataset is termed one *epoch*. More complicated and efficient 'optimizers' have been developed, the Adam optimizer being a famous and commonly used example [123], that improves the method by which the calculated weight gradients are used to update the network weights. The parameters such as learning rate, optimiser momentum, number of layers, number of neurons per layer and so on, are collectively called *hyperparameters*.

1.3.2 Training ONNs: Offline vs Online training

Implementing the backpropagation algorithm, and performing gradient descent in general, is far more difficult using analog hardware compared to digital. In this case, the computer operates using some physical process that is an *analog* of the desired function to be calculated. Therefore, a physical process is needed that behaves just like each particular mathematical function required, and this is not always possible, let alone easy to implement in practice. In particular, the backpropagation algorithm requires calculating the derivative of the nonlinear activation function, given by (1.9). For an ONN, implementing the activation function itself is possible with a range of nonlinear optical phenomena outlined above, but it is extremely challenging to implement both the nonlinearity and its derivative entirely with optics.

Therefore most implementations of ONNs use the optical or photonic hardware just for inference - making new predictions on unseen data [88, 91]. Training is performed entirely separately using a digital simulation, often referred to as offline training [124] or *in-silico* training [125]. The advantage of such a scheme is that the optimal weights can be found using gradient descent and backpropagation exactly as described in the previous section, performed with digital computers that can operate to arbitrarily high precision with no limitation on the available

functions. The pre-trained model weights can then be mapped to the physical system to perform inference.

Hardware that uses physical phenomena to perform computations, including ONNs, will inevitably exhibit imperfections or aberrations that introduce systematic errors. For example, the splitting ratio in an integrated photonic MZI beam-splitter will never be perfectly balanced, or a pixelated liquid-crystal SLM will always exhibit some level of cross-talk between neighbouring pixels. This is particularly problematic for offline training schemes, as the optimal weights found when using a noiseless digital system may not be the optimal weights for the error-prone analog system. Similarly, the optimal weights may be different for separate analog devices that exhibit different levels or type of aberration and error. For this reason many practical demonstrations of offline-trained ONNs perform worse than their digital counterparts [56, 57, 125]. The issue is compounded in deep networks with many layers; small variations between the physical and simulated systems (the so-called 'reality-gap' [126]) can accumulate and grow with every additional layer. A good conceptual illustration of this problem is given by Wright et al. in the supplementary of [125], with a toy example where a relative error of just 0.5% in a simple polynomial function grows to 30% error after 20 composite iterations of the function.

As a potential solution, using 'noise-aware' or 'hardware-aware' training, where the network is trained with a model that includes simulated aberrations and noise equivalent to that measured in the physical system, can partially compensate for this reality-gap [56, 58]. However this requires the ability to accurately model the full system, and carefully characterise the exact nature of any imperfections. This can be extremely difficult for complex nonlinear phenomena, requires a large computational overhead, and would need to be repeated for every individual system. Alternatively, attempts can be made to suppress the noise altogether, using error-correction techniques [78, 81, 82, 99, 127] or optimisation algorithms [98, 128]. However these methods either require significant time and computing resources and would therefore constitute a large overhead on the overall workload of training an

ONN, or they require many additional components, which limits the scalability of such systems in some platforms such as integrated MZI meshes.

1.3.3 Proposals and demonstrations of ONN online training

Another solution to the reality-gap problem is to use the analog hardware itself for training the neural network, and not relying on a digital model. In contrast to offline training, we refer to this as *online* training, also commonly referred to as *in situ* training. A comprehensive review of both offline and online training is provided by Buckley et al. [124]. Here we highlight a few notable proposals and implementations of online training in ONNs.

In all cases, the goal remains unchanged: finding the optimal network weights that maximise the inference accuracy for a given piece of hardware. Gradient descent is still the de facto method to achieve this, although other methods do exist, such as evolutionary models [129]. Additionally, non-standard network architectures such as reservoir computing and spiking networks often require radically different approaches to training, particularly associative learning and Hebbian-like learning algorithms such as spike-timing dependent plasticity [130].

Whilst backpropagation is considered the best way of calculating the gradients needed for gradient descent, it is also not exclusive; given the difficulty of implementing backpropagation in physical systems, many works use alternatives [124]. Examples include direct feedback alignment (DFA) [131, 132], and perturbative algorithms such as the finite difference method [54] or simultaneous perturbation stochastic approximation (SPSA) [61, 133]. All these demonstrations constitute some form of online training, however these techniques are generally considered inferior to backpropagation: DFA uses the error calculated at only the final layer to update all the parameters, and is known to struggle with deep networks [134]. Meanwhile the perturbative methods involve individually or collectively perturbing each parameter and measuring the resulting change to the loss function, in order to approximate the gradient. The exact gradient is only calculated in the limit

of an infinitesimally small perturbation, thereby necessitating a trade-off between training accuracy and the number of epochs required for convergence [135, 136].

Therefore, the primary goal for online training of ONNs is to implement the backpropagation algorithm using the physical system. Techniques where this is done only partially, i.e some but not all steps are performed with the physical system, are often referred to as 'in-the-loop' training methods. In our experiments we demonstrate such a technique which we call 'hybrid training' [137], where at least one full linear layer of neuron interconnection is performed optically in every training iteration. Full details are given in Chapter 3.

A wide variety of similar implementations in various forms have been demonstrated [57, 68, 125, 138–141]. One such demonstration, termed 'physics-aware training', was recently presented by Wright et al. [125], where in-the-loop training was utilised across a variety of analog hardware, including optics. The key idea is to include the physical system in every training iteration, calculating the necessary values in each forward pass with the physical system (just as it would be used for inference), and using a digital model for the backward pass. They demonstrate in-the-loop training can overcome the reality-gap problem and provide much higher classification accuracies than offline training.

Another optical implementation of in-the-loop training was demonstrated by Zhou et al. with a diffractive ONN [57], (although Buckley et al. classify this as 'fine-tune' training.) In particular, three diffractive layers are trained to perform image classification tasks. Measurements from the actual optical system are used to repeatedly update the digital training model for subsequent layers. The classification accuracy was significantly improved, at the expense of many additional training iterations.

Whilst in-the-loop training has clear advantage over offline training, it still heavily relies on digital computation, which may form a bottleneck in future large-scale ONNs, and may not entirely remove the reality-gap problem on models beyond proof-of-principle [134]. The theoretical improvements in speed and energy efficiency, when performing inference with analog instead of digital hardware, are well documented.

However if only inference is optimised, then training will become the dominant cost of operating a neural network. Hence it is essential to extend analog machine learning methods to training. Ideally, the entire backpropagation scheme should be implemented in the physical system with minimal opto-electronic conversion.

Recall backpropagation requires the calculation of two vectors at each layer to find the necessary weight gradient terms: the activations from the forward pass, and error vectors from the backward pass. These error vectors are calculated as

$$\delta_j^{(l)} = \left(\sum_k \delta_k^{(l+1)} W_{kj}^{(l+1)} \right) g'(z_j^{(l)}). \tag{1.11}$$

That is, each error vector requires performing an MVM, and multiplying the result by the derivative of the nonlinear activation function.

Calculating the linear MVM term is (relatively) conceptually straightforward. There have been numerous proposals for achieving this, inspired by early works by Wagner et al. in free-space ONNs [51]. The crucial realisation was that an appropriate physical system can be used to perform both the forward neuron interconnection and the backpropagation MVM, since both processes require using the same matrix. An additional beam of light can be injected to pass back through the system, or the forward signal retroreflected to perform both forward and backward passes. In this sense, the backpropagation algorithm takes on a more literal physical meaning: an optical error signal that physically *propagates backwards* through the system. Recent proposals have also shown how to achieve this in diffractive [112] and integrated MZI mesh [97] ONNs.

In contrast, it is not at all obvious how to use a physical analog system to implement the required mathematical function for the derivative term. Wagner et al. did propose a suitable scheme in their original 1987 work, suggesting Fabry–Perot etalons as the nonlinear activation function would enable the appropriate derivative term to be calculated optically [51].

In regards to practical demonstration, Psaltis et al. partially demonstrated their original optical training scheme experimentally in 1988 [52], and Pai et al. recently presented results for implementing their in-situ backpropagation scheme in an MZI

mesh ONN [142]. This scheme was particularly novel in being able to calculate the gradients of the loss function with respect to the actual control parameters in the MZI mesh (e.g. phase shifter values), and not just the weight matrix elements. However all experiments to date have failed to implement physical backpropagation through the nonlinear activation function. Psaltis et al. opted to simulate the nonlinear function digitally [52], and the recent work by Pai et al. [142] required calculating the derivative term digitally, and applying it to the system electro-optically.

Therefore in practice end-to-end optical backpropagation is an outstanding challenge yet to be achieved, and in this thesis I present the work undertaken to experimentally implement such a scheme, including both the linear interconnection and nonlinear activations, to successfully train a multi-layer ONN. My experiment builds on earlier theoretical and simulation work from our group [143], which shows that saturable absorption is a suitable optical phenomena that allows backpropagation through the nonlinear activation function. Full details of this theoretical proposal and our experimental results are given in Chapter 5.

The motivation for using novel analog hardware for neural networks, such as in ONNs, is the potential benefit provided in speed and energy efficiency over digital hardware. To fully benefit, it will be necessary to use such hardware to perform all aspects of neural networks: linear neuron interconnection, nonlinear activation, training and inference. Our work has been focused on achieving exactly this, to demonstrate it is feasible to perform all stages of a neural network entirely with optics.

2

Optical Matrix-Vector Multiplication

Contents

This chapter outlines the concepts, experiments and results of our optical matrix-vector multiplier, which can perform multiplications at large scale, using coherent light, in such a way that the vectors and matrices can be arbitrarily reconfigured. Some of the results and methods in this chapter are presented in the manuscript '*Fully reconfigurable coherent optical matrix-vector multiplication*' [96] (Author contribution: joint-first author, carried out experiments and performed data analysis. Equal contribution to manuscript preparation.)

2.1 Concepts and Theory

2.1.1 Conceptual scheme

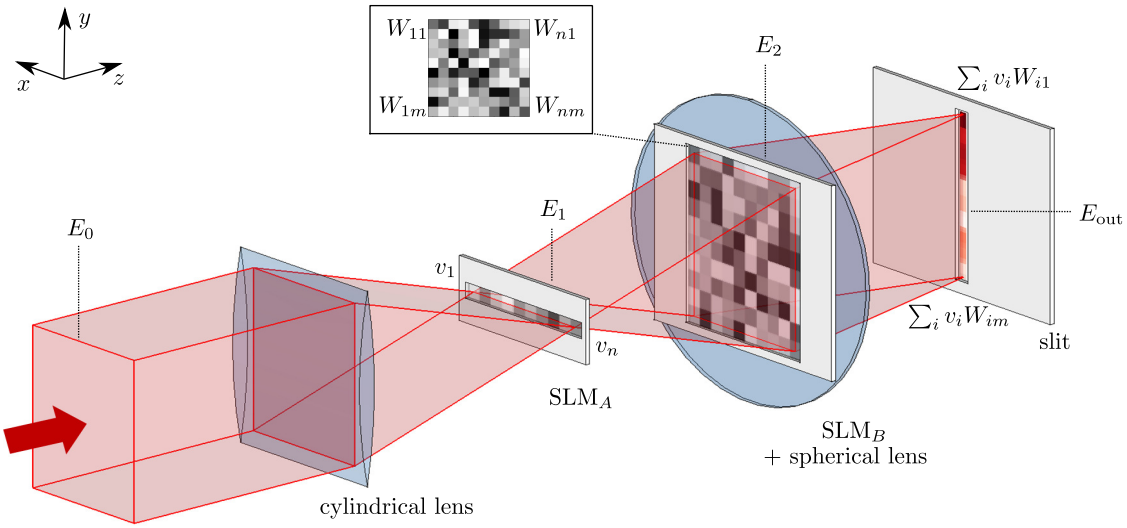


Figure 2.1: Conceptual diagram of coherent optical MVM using two spatial light modulators, lenses and slit.

As already motivated in the introduction, matrix-vector multiplication (MVM) is the core operation of most neural networks, and can be implemented in many optical setups. We begin by outlining how free-space optics can be used to perform real-valued MVM, following known schemes from many important early studies of optical computing [46, 49]. Fig. 2.1 depicts the concept. The vector and matrix elements are encoded in the spatial structure of the electric field amplitude of a monochromatic coherent laser beam. By encoding the elements with the correct geometry, and manipulating the structure of the beam with lenses, the multiplication occurs passively as the beam propagates through the system. We encode the vector and matrix elements using a spatial light modulator (SLM), which here we give as a catch-all term for any device that can spatially vary the amplitude, phase or intensity of a light beam. SLMs take many forms, each with different benefits and drawbacks, but for this conceptual description we take them to be ideal modulators, capable of arbitrarily varying the amplitude and phase at any transverse position of the beam.

We want to perform the real-valued calculation $\vec{u} = \boldsymbol{W}\vec{v}$, where $\boldsymbol{W}$ is an $M \times N$ matrix and $\vec{v}, \vec{u}$ are vectors of sizes N and M, which we index with i and j respectively. The MVM can be written as

$$u_j = \sum_{i=1}^{N} W_{ji} v_i. \tag{2.1}$$

Consider two ideal SLMs, SLM_A and SLM_B with spatial coordinates (x, y). SLM_A is one-dimensional with physical size L_x. SLM_B is two-dimensional with physical size $L_x \times L_y$. They are designed to have complex field transmissions

$$v(x) = v_i \tag{2.2}$$

$$W(x, y) = W_{ji}, \tag{2.3}$$

where

$$i = \left\lceil \frac{Nx}{L_x} \right\rceil, \quad j = \left\lceil \frac{My}{L_y} \right\rceil.$$

A coherent light source producing a uniform electric field E_0 is focused in the y-direction onto SLM_A by a cylindrical lens, to produce a narrow, horizontal field

$$E_1(x, y) = E_0 v(x) \delta(y). \tag{2.4}$$

This field spreads vertically onto SLM_B, a process we refer to as 'fan-out', so the field immediately after SLM_B can be written as

$$E_2(x, y) = E_0 v(x) W(x, y), \tag{2.5}$$

equivalent to element-wise multiplication between the vector and each row of the matrix. To sum these outputs over index i the field is focused in the x-direction, a process we refer to as 'fan-in'. This is equivalent to a Fourier transform in the x-direction, so the field at the output is

$$\begin{aligned} E_2(k_x, y) &= \mathscr{F}_x \left(E_1(x, y) \right) \\ &= \frac{E_0}{2\pi} \int v(x) W(x, y) \exp\left(\frac{i k_x x}{\lambda f} \right) dx. \end{aligned} \tag{2.6}$$

Finally we pass the beam through a narrow slit, selecting just the zero-order component $k_x = 0$. By assigning

$$u_j = E_2\left(0, \frac{jM}{L_y}\right), \tag{2.7}$$

the resulting field exactly encodes the desired MVM result, Eq. (2.1), up to some global scaling factor:

$$u_j = \frac{E_0}{2\pi}\int v(x)W(x,y)dx \tag{2.8}$$

$$\propto \sum_i v_i W_{ji}. \tag{2.9}$$

Coherent detection of the field at this plane allows the MVM result to be extracted from the beam.

As mentioned in the introduction, the principle of conservation of Étendue means only a fraction of the power is admitted at the output of the system after optical fan-in. This is seen explicitly here by the application of the narrow slit, which must have a width proportional to the matrix size in order to admit only the $k_x = 0$ component, hence the power transmitted by the MVM scales proportionally as $1/N$.

We can summarize the necessary steps for optical MVM as follows:

- Vector-encoding and fan-out,
- Matrix-encoding,
- Fan-in and detection.

The scheme we use in experiment follows the principles outlined above, but with different practical implementations. The following sections outline in detail how each of these steps were implemented.

2.1.2 Vector-encoding and fan-out: DMD

The type of SLM used to encode the input vectors was chosen to be a digital micro-mirror device (DMD). Among other applications, these MEMS devices are primarily used in Digital Light Processing (DLP) projectors [144]. A beam of

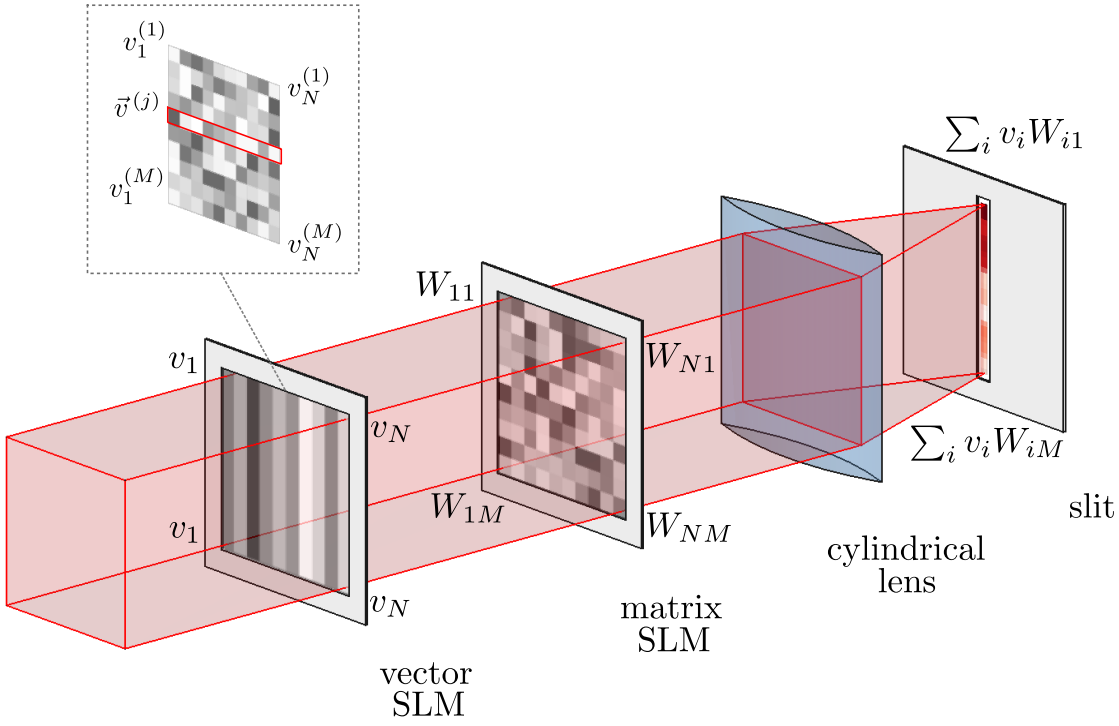


Figure 2.2: Scheme of coherent optical MVM implemented in experiment. The full 2–D DMD plane is used to simultaneously encode the input vector and perform 'fan-out'. Reprinted with permission from [96] © Optica Publishing Group.

light is reflected from a small, pixelated, reflective window, where each pixel is an independently-addressable mirror with two states, 'on' and 'off'. When all mirrors are set to 'on', the DMD acts as a plane mirror, but any pixels that are turned 'off' deflect that region of the beam away from the signal path. We want to encode positive-only, multilevel values for the vector input, and this can be achieved by grouping many physical, binary pixels into groups of 'logical pixels'. A fraction of physical pixels within each logical pixel are turned on to create a block, and by varying the width of the block values between 0 (all pixels off) and 1 (all pixels on) can be encoded. The number of numerical values that can be encoded is determined by the number of physical pixels per logical pixel in the 'width' dimension, which must be the dimension in which the cylindrical lens acts to converge the beam. The result is a multi-level analog encoding of our input vector, and given the fixed resolution of the DMD, there is a necessary trade-off between maximum vector dimension and the equivalent bit-depth of the encoding.

The first major difference between the conceptual scheme and our experiment setup is the fan-out procedure. Instead of encoding a 1D vector of dimension N and spreading the beam with a cylindrical lens, we utilise the full 2D area of the DMD and directly encode M copies of the vector, creating an $N \times M$ array $\left\{\vec{v}^{(1)} \dots \vec{v}^{(M)}\right\}$. This 2D image is then mapped directly to the matrix-encoding SLM. This scheme is shown in Fig. 2.2, and has multiple benefits; we eliminate the need to spread the beam with a cylindrical lens, which would add some degree of aberration to the beam, and it allows the flexibility of encoding different vectors at each vertical position. This can be used to compensate for a non-uniform input beam for example. It also allows us to generalise matrix-vector multiplication (MVM) to parallel vector-vector multiplication (p-VVM). MVM is a special case of p-VVM when all the input vectors are the same. The inset of Fig. 2.2 depicts an alternative example encoding of M different vectors of dimension N.

2.1.3 Matrix-encoding: LC-SLM

Phase-only nematic liquid-crystal-on-silicon spatial light modulators (LC-SLMs) are extremely useful devices, utilised across a wide variety of applications, such as optical tweezers [145], wave-front shaping [146], communications [147] and adaptive optics [148]. Like the DMD, it is usually a reflective, pixelated device, although transmissive versions also exist. A layer of nematic liquid-crystal is held between one electronically-addressable, pixelated electrode (the backplane), and a transparent front electrode. Applying a voltage across any individual pixel locally rotates the liquid crystal molecules. Due to the birefringence of the material, this causes the refractive index to change, and so a variable phase delay can be imparted to the beam being reflected. The range of voltage required to induce a full 2π phase shift is denoted $V_{2\pi}$, and LC-SLMs can have up to 10-bit resolution across this range.

In practice LC-SLMs are not ideal devices, and require substantial calibration to correct for nonlinear and nonuniform phase response that arise from various sources [149–152]. Furthermore LC-SLMs are phase-only modulators, so cannot directly modulate the amplitude of a beam. However various techniques have been

developed to allow simultaneous and independent control of both the amplitude and phase of a coherent beam [153–155]. The technique we employed is originally attributed to Bolduc et al. [156], and is summarised as follows.

We want to encode in our beam the complex field

$$s(x, y) = A(x, y)e^{\phi(x,y)}. \tag{2.10}$$

To achieve this the SLM should generate a phase profile $\psi(x, y)$ in the form of a phase grating. Correctly modulating the height and offset of the grating can modulate the amplitude $A(x, y)$ and phase $\phi(x, y)$ of the output field. The grating can take different forms [157], and in this work we choose to use a diagonal blazed grating, so that the SLM phase profile is given by

$$\psi(x, y) = M(x, y)\mathrm{mod}_{2\pi}\left[F(x, y) + u_0 x + u_0 y\right], \tag{2.11}$$

where u_0 is an arbitrary parameter to be chosen, and two corrective terms are applied to the grating: the function

$$M(x, y) = 1 - \mathrm{sinc}^{-1}[A(x, y)] \tag{2.12}$$

compensates for an amplitude factor introduced when taking the Fourier transform of a blazed grating. It includes the inverse sinc function, where $\mathrm{sinc}(x) = \sin(\pi x)/\pi x$. A similar corrective term

$$F(x, y) = \phi(x, y) + (1 - M(x, y))\,\pi \tag{2.13}$$

must then also be applied to maintain the desired phase profile.

This phase grating will produce multiple diffraction orders, and by using a spatial filter that admits only the first diffraction order at the Fourier plane, an image with the correct amplitude and phase can be recovered. The spatial filter comprises two spherical lenses and an iris in the Fourier plane centred on the first diffraction spot. It is non trivial to prove this is the required form of $\psi(x, y)$ to correctly reproduce the target amplitude and phase - details are given in Appendix B.

Choosing the parameter u_0 in (2.11) is significant: when we spatially filter the beam in the Fourier plane, the maximum spatial frequency bandwidth is

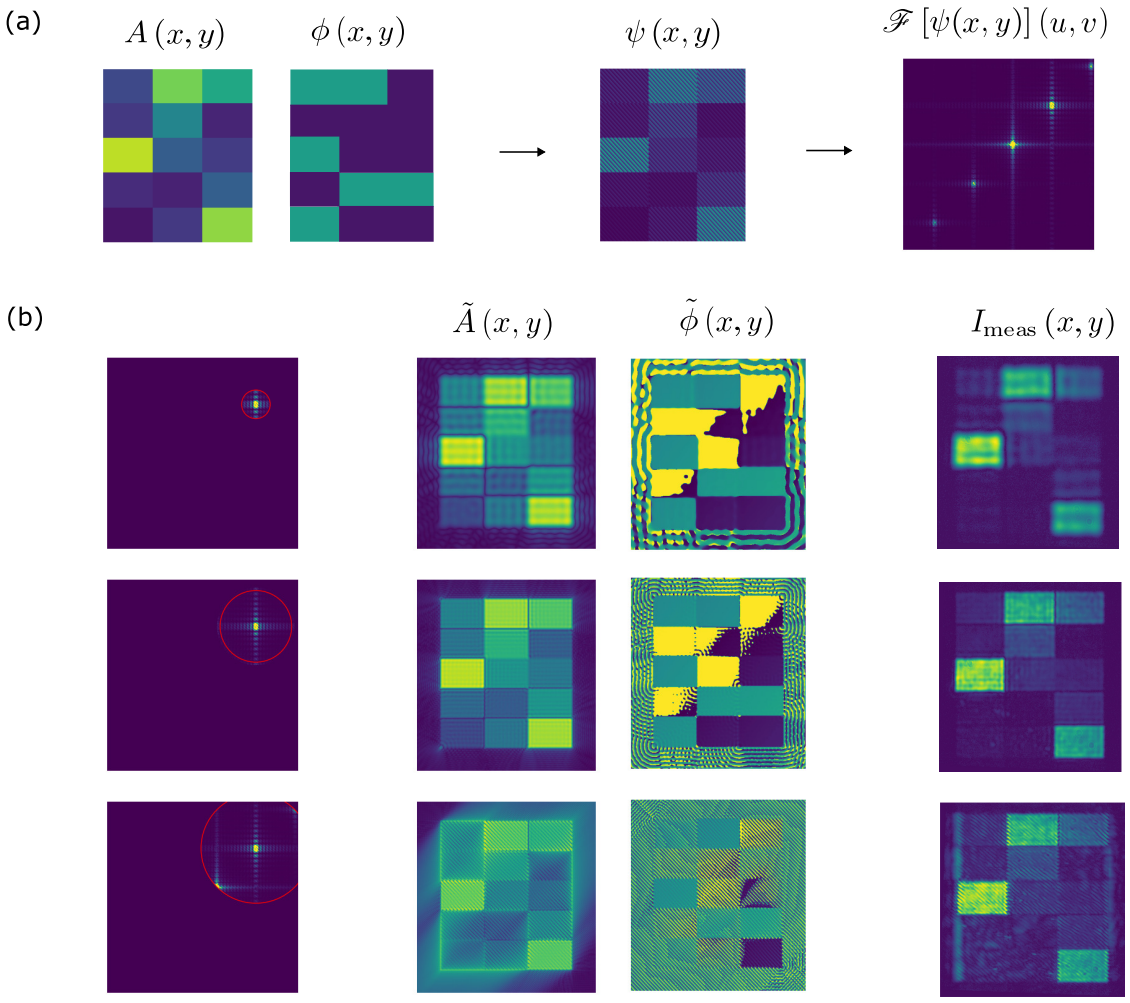


Figure 2.3: Simulation to demonstrate the phase grating technique for real-valued encoding with an LC-SLM. (a) Example amplitude and phase patterns for a random matrix, associated LC-SLM phase profile, and simulated Fourier transform. Note the diagonal phase grating pattern corresponds to the diagonal diffraction spots in the Fourier plane. (b) Simulated amplitude and phase profiles of the reconstructed image after inverse Fourier transform of the first diffraction order, with different sizes of iris. Corresponding intensity images from experiment are also shown.

bounded by u_0, which then determines the finest level of detail achievable in the output image. In fact

$$u_0 = \frac{2\pi}{p}, \tag{2.14}$$

where p is the period of the grating we display on the SLM. Therefore a smaller grating period allows a greater bandwidth and finer detail of the final image, at the cost of a less efficient phase grating. In practice the chosen grating period is fixed,

and the size of the iris at the Fourier plane is tuned to maximise the bandwidth, without admitting additional diffraction orders.

A simple digital simulation of this phase grating technique was coded in Python. Fig. 2.3(a) shows an example target amplitude and phase pattern, resembling the patterns required to encode a real-valued random 5×3 matrix with amplitude $A(x, y)$ and phase $\phi(x, y)$. Each matrix element is encoded by a block of many pixels. The phase pattern $\psi(x, y)$ calculated using (2.11) is also shown, as is the simulated 2–D Fourier transform of the complex field profile that would be generated by passing the beam through a spherical lens. This represents the field that would be seen at the Fourier plane of the spatial filter, with the different diffraction orders clearly visible.

The effect of the iris size is explored in Fig. 2.3(b), for the same example patterns used in (a). We simulate the effect of an iris positioned at the first diffraction order, by setting to zero all values outside a set radius r, as shown in the left column, for three empirical values of r. Taking the inverse 2–D Fourier transform of this 'masked' field simulates the complex output field of the spatial filter, and the amplitude and phase of this output field is shown. When the radius is too small (top row), higher order components are lost, and the resulting image is significantly blurred. In contrast when the radius is too large (bottom row), the zero diffraction order is admitted, which adds high frequency noise to the image. When the radius is adjusted suitably (middle row) the target amplitude and phase is well reconstructed, except for arbitrary jumps of 2π in the phase profile that can be safely ignored.

The final column of Fig. 2.3(b) shows intensity $I_{\text{meas}}(x, y)$, equivalent to $|A(x, y)|^2$, measured in experiment, when an LC-SLM and spatial filter is used to produce the example matrix. The iris is adjusted to demonstrate the three cases explored in simulation, and we see excellent agreement between theory and experiment.

2.1.4 Fan-in and detection: Cylindrical lenses and CCD

The fan-in process is performed by a set of three cylindrical lenses, equally spaced between the input and output planes. Fig. 2.4 shows the geometry of the beam as is passes through the lens system. Crucially, the central lens L_2 is rotated by 90° and

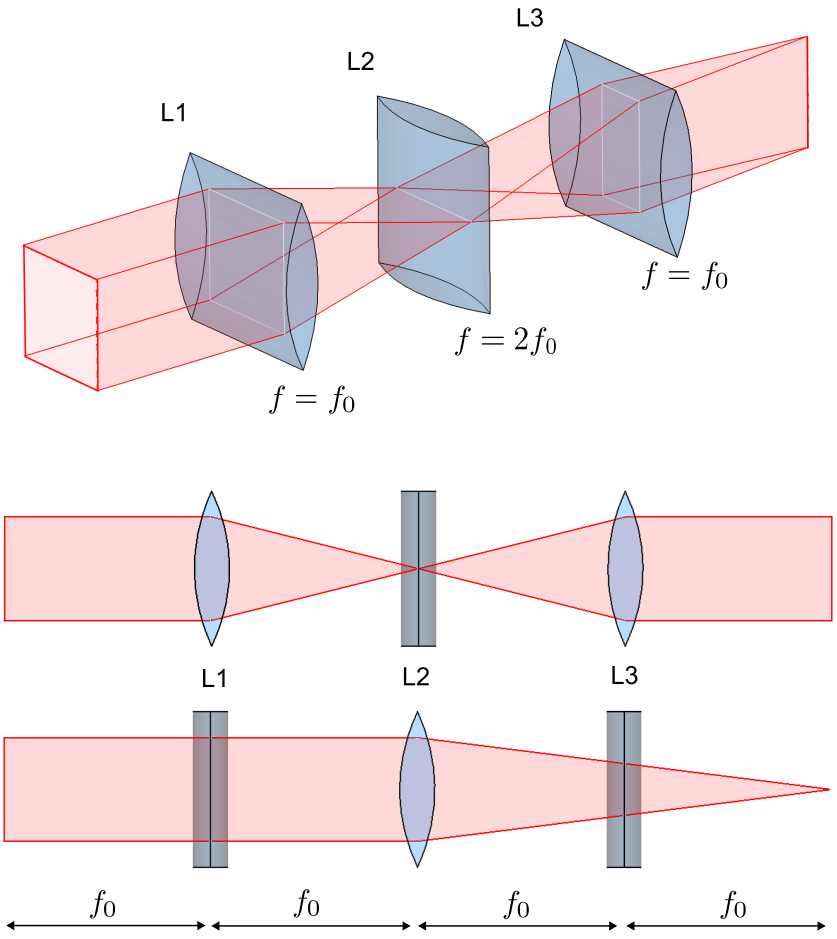


Figure 2.4: The geometry of the beam as it passes through the three cylindrical lenses used to perform optical fan-in. All three lenses are equally spaced a distance f_0 between input and output planes. Lenses L_1 and L_3 have focal length f_0 and form a $4f$ imaging system in the vertical dimension. Lens L_2 has focal length $2f_0$ and performs Fourier transform in the horizontal direction to complete optical fan-in.

has a focal length twice as long relative to the outer lenses L_1 and L_3. The effect is for L_2 to perform a Fourier transform in the horizontal x direction as required, whilst the outer lenses act as $4f$ system in the vertical y direction, preventing unwanted diffraction and maintaining the correct phase and amplitude profile.

In this experiment, we avoid the need for a narrow slit to select the zero-order component of the output field by detecting the beam using a high resolution CCD camera. The entire output plane is imaged by the camera, and the region that corresponds to the zero-order component is digitally selected.

2.2 Experiment

2.2.1 Methods

A full experiment schematic and simplified diagram are shown in Fig. 2.5, combining the three techniques outlined above. A laser beam from an External Cavity Diode Laser (ECDL), model Toptica DL100, is used as the coherent beam source, producing

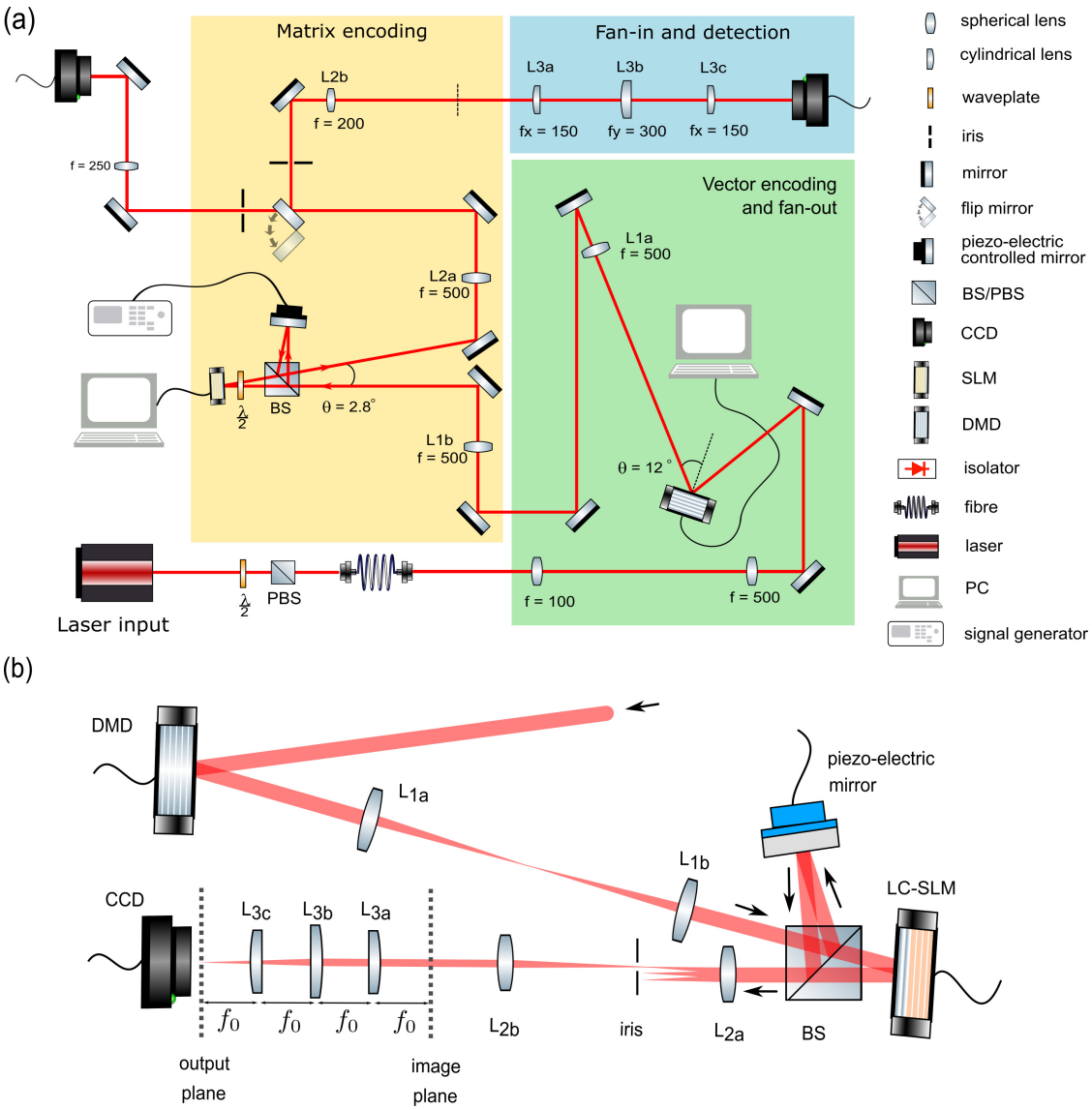


Figure 2.5: Diagrams to show experiment setup for optical MVM. (a) Full experiment diagram, showing how the stages of vector encoding, fan-out, matrix encoding, fan-in and detection are combined to perform optical MVM. (b) Simplified diagram, showing the interferometer used to perform coherent detection. Adapted with permission from [96] ©Optica Publishing Group.

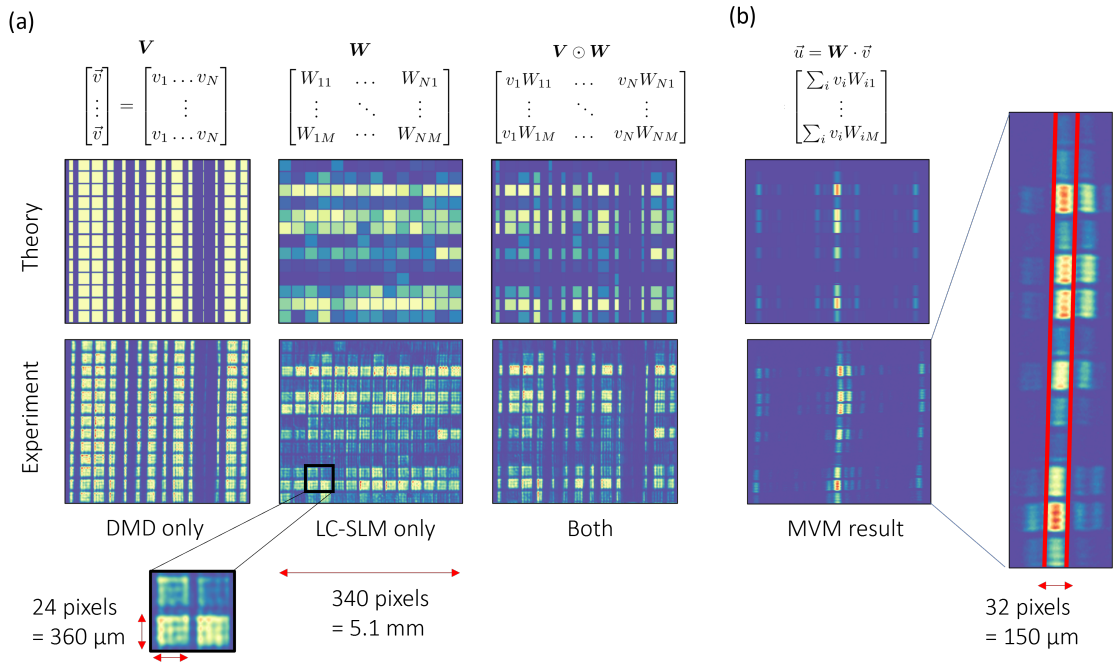


Figure 2.6: Theoretical and experimental images of the fields created by the DMD and LC-SLM, and the associated MVM result, for one example with random vector and matrix elements. (a) Images recorded with the camera at the image plane before the cylindrical lenses, of the DMD only, LC-SLM only and both, representing their element-wise product. (b) MVM results are extracted from the intensity image in the output plane. The simulated image and experiment image recorded for the example in (a) are shown. Figure adapted with permission from [96] ©Optica Publishing Group.

a narrow-linewidth 40 mW beam at 780 nm. A Tapered Amplifier (TA) is used to amplify the intensity, and the beam is coupled through a polarisation-maintaining fibre to produce a beam with a high quality Gaussian profile. The maximum power at the output of the fibre was approximately 300 mW. The power could be variably attenuated using the combination of a half-waveplate and polarising beam splitter.

The beam was broadened with a pair of spherical lenses to produce a large-waist beam, and centred onto the DMD, model TI Discovery 1100. The DMD encodes the vector input as described previously. A fixed area of the DMD was used as the 'signal' region, measuring 380×380 pixels with a pixel pitch of 13.68 μm, chosen to match the size of the LC-SLM active region. An $N \times M$ array of logical pixels was encoded, each formed from a $p \times p$ square block of physical pixels. Recall a block of physical pixels with variable width is turned on to encode multi-level inputs, with $p + 1$ levels. The value of p depends on N, but was a minimum of $p = 3$ for

$N = 56$ (in which case the values $0, 0.33, 0.66, 1$ could be encoded.)

After reflection from the DMD, the beam is mapped to the LC-SLM, model Meadowlark P512, using a pair of spherical lenses with focal length 500mm, L_{1a} and L_{1b}, forming a $4f$ imaging system. The LC-SLM produces minimal aberration when reflecting at 0° angle, so the reflection angle was kept as small as possible. The spatial filter after the LC-SLM comprised two spherical lenses L_{2a} and L_{2b}, and an adjustable iris centred at the first diffraction order. The LC-SLM used in this experiment was an old device, and the useable active area had a limited pixel resolution of approximately 340×340, with pixel pitch of 15μm. The phase grating had a period of four pixels, and as previously explained the LC-SLM grating encoding method can only produce images with structure greater than the grating period. Therefore each matrix element was encoded by a group of pixels at least 5 pixels square.

After the spatial filter, a cylindrical lens set L_{3a}, L_{3b} and L_{3c} performs the fan-in process, and the entire output plane is measured by a CCD camera, model Ueye UI-2230SE-M-GL. Lens L_{3b} has focal length $2f_0 = 300$mm whilst L_{3a} and L_{3c} have focal length $f_0 = 150$mm. The camera has pixel resolution 1024×768, with a pixel pitch of 4.65μm.

Figure 2.6(a) shows simulated and experiment images of the field generated by the DMD and LC-SLM. The experiment images are taken by a second CCD camera in an alternative optical path, accessed by a flip mirror used to bypass the cylindrical lenses and directly image the LC-SLM (as shown in Fig. 2.5), which is also at the image plane of the DMD. The first image is taken by displaying a random vector of size $N = 14$, in the fanned-out $N \times M$ array, whilst keeping the LC-SLM target pattern entirely uniform. The second image shows the reverse, with the DMD kept uniform whilst the LC-SLM displays a random $N \times M$ matrix. The third images shows the combination when both random vector and matrix are displayed, corresponding to their element-wise multiplication.

Figure 2.6(b) shows the final MVM output field for the random vector and matrix patterns in (a). The simulated image is the horizontal Fourier transform

of the element-wise multiplication field, and the experiment image is the intensity recorded by the camera after the cylindrical lenses. The output field is a complex interference pattern, but only the field along the narrow central region encodes the MVM result. This area is indicated in Fig. 2.6(c) and is 32 pixels, or approximately 150μm, wide. The MVM result is extracted by digitally processing the measured pixel values in this region.

In all these images we see very good agreement between theory and experiment. However the output plane clearly shows aberration, with a curved and non-symmetric pattern. This is attributed to a non-uniform phase profile generated by the LC-SLM, and imperfect cylindrical lenses. Importantly, throughout this experiment the aberration did not change, so the zero-order region remained fixed and no additional digital processing was required. In later work, different LC-SLM models were used, as well as higher-quality cylindrical lenses, and this aberration was significantly reduced.

2.2.2 Coherent detection

Since we perform real-valued MVM, the output result contains amplitude and phase information. We label this field

$$E_{\text{sig}} = A_{\text{sig}} e^{i\phi_{\text{sig}}}.$$

In order to measure the phase, we first create an interferometer with a beam-splitter and piezo-electric mirror next to the LC-SLM, generating a uniform reference beam

$$E_{\text{ref}} = E_0 e^{i\phi(\delta)}.$$

A function generator is then used to scan the voltage applied to the piezo, which in turn modulates the reference beam delay length δ. The camera measures the intensity of the coherent sum of the signal and reference beams:

$$\begin{aligned} I_{\text{meas}}(\delta) &= |E_{\text{sig}} + E_{\text{ref}}|^2 \\ &= |A_{\text{sig}} e^{i\phi_{\text{sig}}} + E_0 e^{i\phi(\delta)}|^2 \\ &= |A_{\text{sig}}|^2 + |E_0|^2 + 2E_0 A_{\text{sig}} \cos\left(\phi_{\text{sig}} - \phi(\delta)\right). \end{aligned} \tag{2.15}$$

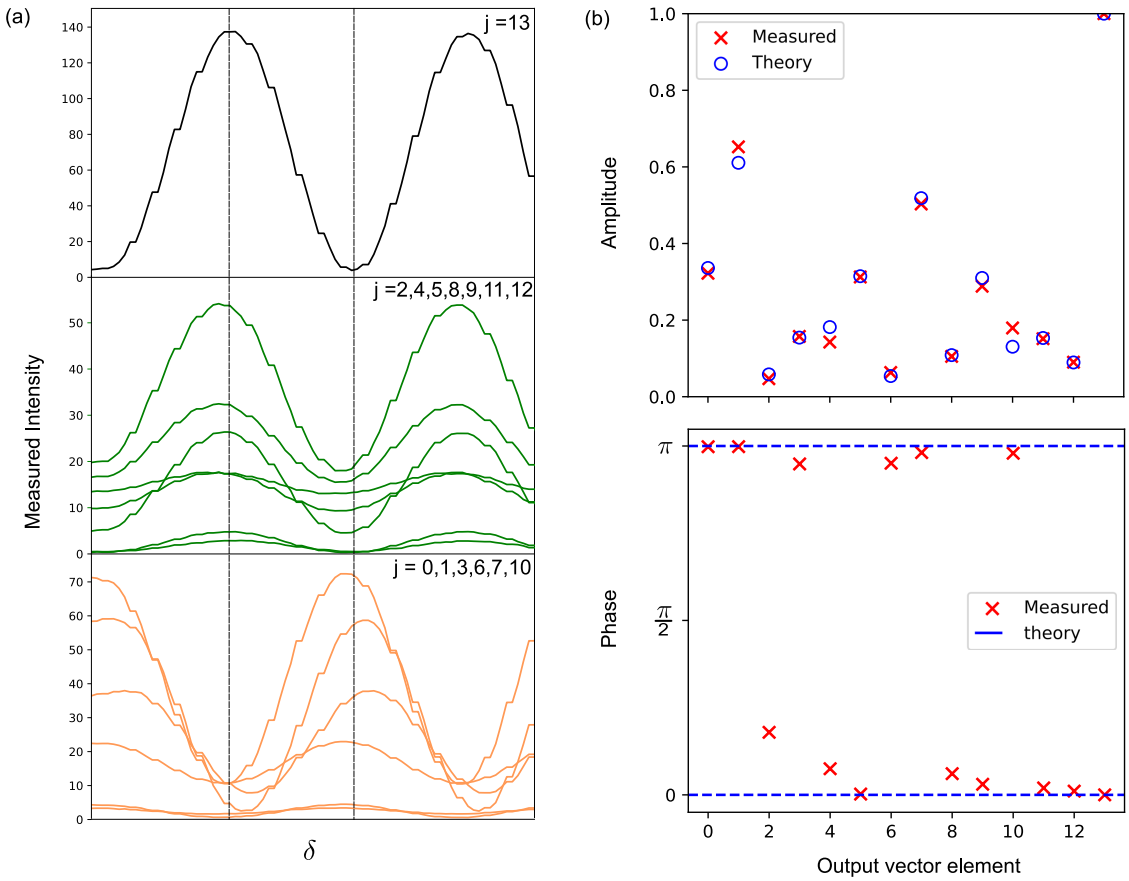


Figure 2.7: Results for one example of optical MVM with random vector and matrix at size $N = 14$. (a) Intensity recorded by the camera for each vector element, as the reference phase is modulated by the piezo mirror. Plots are separated by reference element kept at maximum brightness (top panel), theoretical values of zero phase (middle panel) and π phase (bottom panel) (b) Measured amplitude and phase values of each output vector element, extracted from the associated intensity curve (red crosses), as well as the theoretical MVM result (blue circles / dashed line).

Figure 2.7(a) shows an example of the intensity scans recorded for each output element of one MVM result. In every multiplication, the first row of both the fanned-out-vector and the matrix is kept maximally uniform, to provide a reference output element that has constant phase and amplitude. The top panel shows the intensity scan for this reference output element. The middle panel shows the intensity scans for all elements that are positive (that is, have theoretically zero phase), and bottom panel shows all negative elements (theoretically π phase). The measured positive and negative elements are clearly distinguished by a π-phase shift.

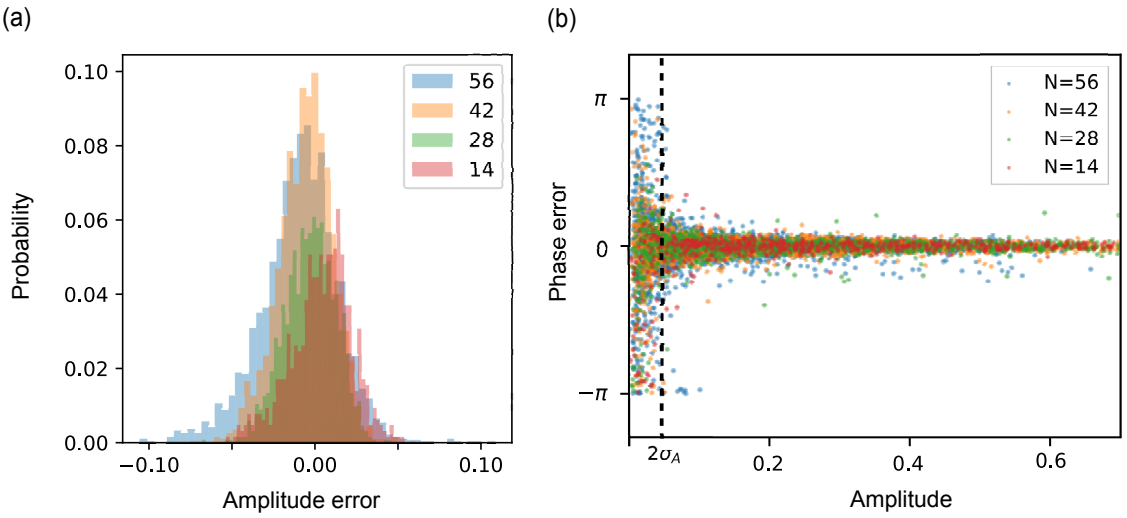


Figure 2.8: Results for many repeated measurements of optical p-VVM. (a) Histogram for the errors in amplitude between measured and theory values, for each dimension. (b) Phase error between measured and theory values, as a function of measured amplitude, for each dimension. The value of $2\sigma_A$ for $N = 56$ is indicated by the dashed line.

By fitting a curve of the form

$$f(x) = a + b\cos(x + c) \tag{2.16}$$

to the intensity scan, we can use the fitting parameters to extract the amplitude and phase of each output element. Figure 2.7(b) plots the measured amplitude and phase of each vector element intensity scan in (a), as well as the theoretical (digitally calculated) result of the MVM. The amplitude values are normalised such that the reference row (kept at maximum brightness) is equal to one, and we see good agreement between theory and experiment. We quantify the precision of our multiplier by performing many such example measurements, recording intensity scans, and numerically extracting the amplitude and phase.

2.2.3 Results and analysis

Recall p-VVM is achieved by setting different vectors on each row of the DMD, and is a more general version of MVM. In fact p-VVM is likely to show larger error, as it requires precise alignment between the SLMs in both horizontal and vertical directions. We perform a batch of 50 such p-VVMs for vector sizes $N = 14, 28, 42, 56$. We use square matrices, but taking account for the reference row we have $M = N - 1$.

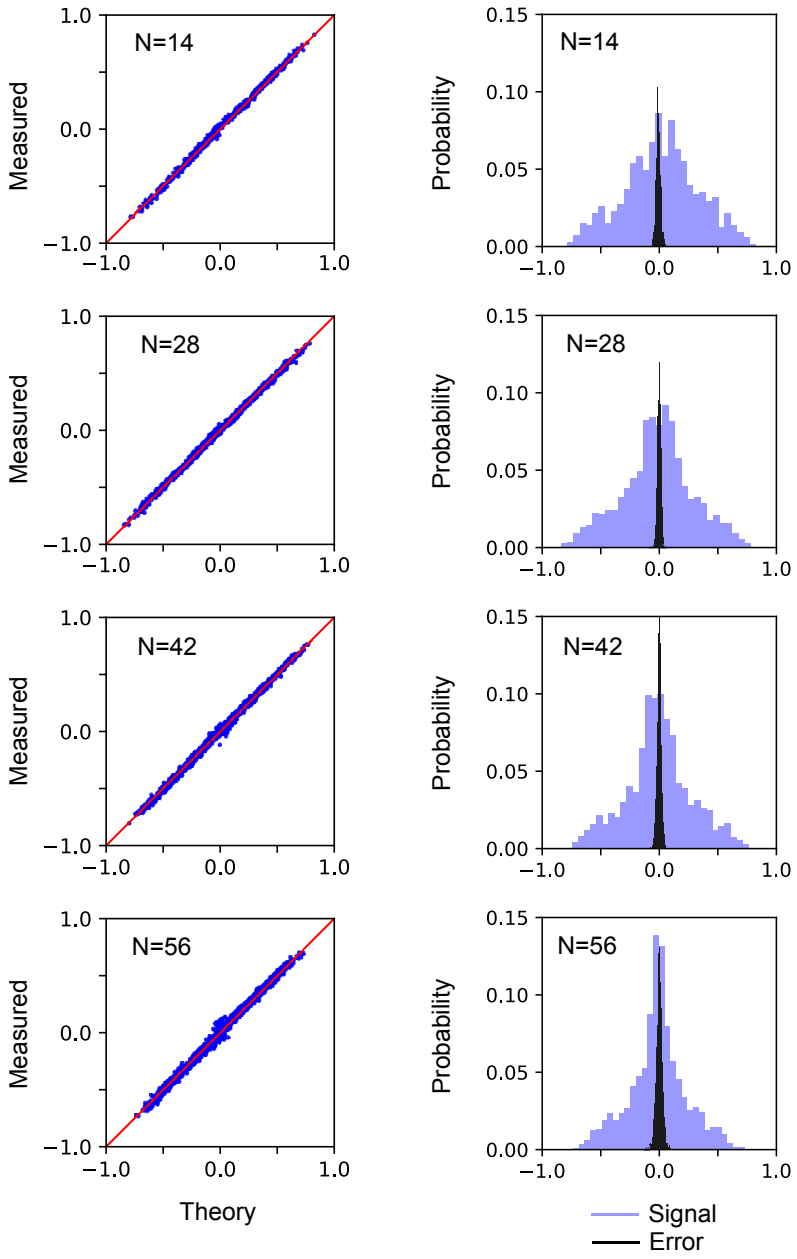


Figure 2.9: Individual scatter plots for the 50 repeated measurements of optical p-VVM, for dimensions N=14,28,42,56. Left hand panels show scatters of measured against theory values for the real-valued results, for each dimension. Right hand panels shows the histogram of the theoretical result (signal, blue) compared to the calculated error (black). The ratio of the widths of these distributions gives the SNR.

In every case the vector elements are chosen at random in the range $[0, 1]$, and matrix elements in range $[-1, 1]$. The output vector values are normalised by the reference row output, which is the maximum possible output value, so that the outputs have amplitude normalised in the range $[0, 1]$, whilst the phase is

measured in the range $[-\pi, \pi]$.

The analog nature of our optical multiplier means the multiplication result will inevitably exhibit some level of noise. We quantify this noise by measuring the root mean square error (RMSE) between our multiplier output and the equivalent digitally-calculated result, when both are normalised as described above. First, we measure the RMSE of amplitude, and plot the amplitude error distributions in Fig. 2.8(a). The standard deviations of these error distributions are $\sigma_A = 17.9 \times 10^{-3}, 15.1 \times 10^{-3}, 16.4 \times 10^{-3}, 23.2 \times 10^{-3}$ for $N = 14, 28, 48, 56$ respectively.

As demonstrated by the previous example in Fig. 2.7, the phase is consistently measured to be close to 0 or π, although there are some elements with larger phase error. In Fig. 2.8(b) we plot the phase errors as a function of amplitude for all dimensions, and we see the majority of values with large phase error occur when the amplitude is small, due to the poorer numerical fit of the interference pattern. These larger phase errors are therefore not a concern. For example, for dimension $N = 14$, the standard deviation of phase errors is only $\sigma_\phi = 64.6 \times 10^{-3}\pi$ when considering points with amplitude $A > 2\sigma_A$.

We use the measured phase and amplitude to produce real-valued outputs in $[-1, 1]$ by rounding the phase to either zero or π. We then define the signal-to-noise ratio (SNR) as the ratio of the standard deviation of the signal range σ^N_{sig} and the measured RMSE σ^N_{err}. Figure 2.9 shows our multiplier has very high SNR: for each dimension, we plot a scatter of the measured and theory values, for all 50 examples and every output element. Every scatter point should fall along the red diagonal line, and the level of noise can be visualised by the width of the resulting scatter plot. Below each scatter we plot histograms of the measured signal value (blue) and the calculated errors (black), and the ratio of their widths gives the SNR. For $N = 14, 28, 42, 56$ we find $\sigma^N_{\text{err}} = 17.0 \times 10^{-3}, 15.5 \times 10^{-3}, 16.9 \times 10^{-3}, 23.7 \times 10^{-3}$ respectively. The corresponding SNRs are 18.6, 19.5, 16.5 and 10.2. At size $N = 56$ each logical-pixel is only modulated by one grating period, so the diffraction efficiency is minimal, and we see a significant drop in multiplier precision at this size. At smaller sizes, the multiplier precision is limited by the LC-SLM

calibration, which is independent of dimension. The calibration procedures are detailed in the next section.

2.3 Calibration and system improvements

Extensive effort was made to calibrate our system to minimise the error of the MVM output, whilst still allowing the system to be rapidly reconfigured, without the need to use complicated and time-consuming iterative algorithms. The MVM precision is primarily limited by the fidelity of the complex field being generated by the LC-SLM. Some of the techniques used to maximise this fidelity are described below.

Additionally, modern LC-SLM and DMD models offer greater pixel resolution, better reflection efficiency, more linear response profiles, and smaller phase profile distortion. After conducting the initial MVM experiment, our optical multiplier was significantly improved by replacing the LC-SLM and DMD. With a greater pixel resolution, the achievable vector and matrix size was able to increase to nearly 200×50. The details and results of this improved optical multiplier are described in detail in 3.1.

2.3.1 Voltage-phase response of the LC-SLMs

In an LC-SLM, the local phase delay caused by the liquid crystal is a function of the voltage applied at each pixel. But this response is different for every pixel (non-uniformity), and is often not a linear mapping between voltage and phase-delay (non-linearity). Therefore, we must measure the voltage-phase response across the LC-SLM, and generate a look-up-table (LUT) of required pixel voltage for the desired phase delay.

To measure this response, we utilise the fact that LC-SLMs have a specific working polarisation. This polarisation dependence is illustrated in Fig. 2.10. When a voltage is applied across the liquid crystal, controlled by an 8-bit Grey-level (GL), it has the effect of rotating the optic axis of the material within a fixed plane. If the incoming light is linearly polarised in this plane, as in Fig. 2.10(a), the changing optical axis of the liquid crystal changes the effective refractive index,

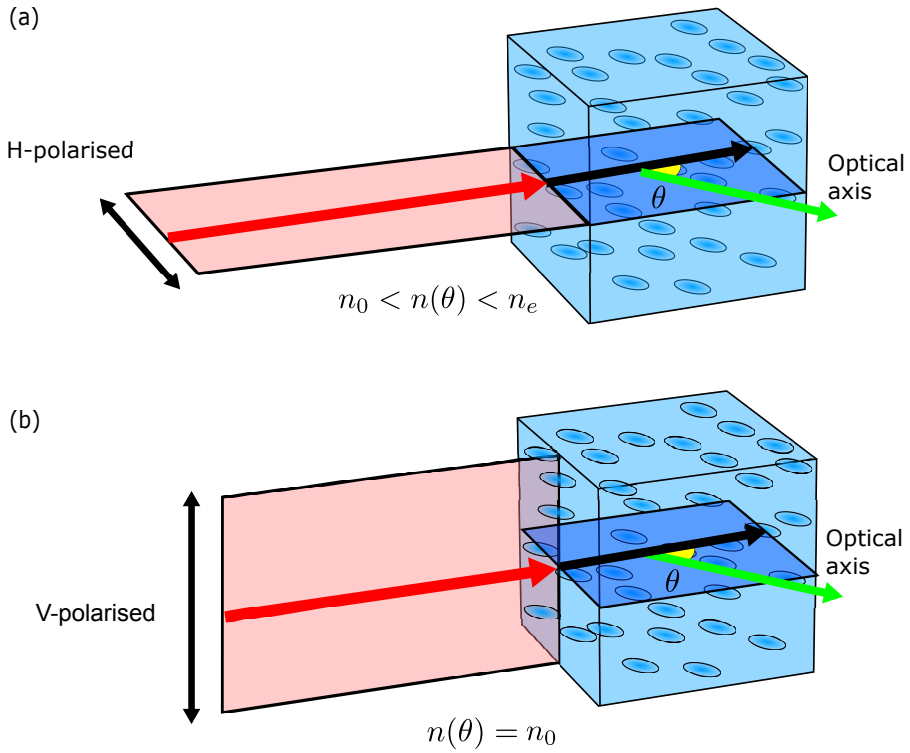


Figure 2.10: The LC-SLM phase delay is modulated by rotating liquid crystal molecules, and requires the incident light to be polarised in the correct plane. (a) If polarised in the plane of optical axis rotation, the refractive index is dependent on the angle θ. (b) If polarised perpendicular to the plane of rotation, the refractive index remains constant.

$n(\theta)$, between the limits of ordinary and extraordinary refractive index values, n_o and n_e. This in turn modulates the phase delay of the light. However, if the light is polarised orthogonal to the optic axis plane, as shown in Fig. 2.10(b), the refractive index remains fixed as n_o, and the phase delay is constant and independent of liquid crystal orientation.

Therefore in normal operation we position a half-waveplate before the LC-SLM to rotate the polarisation of the beam to lie in the optic axis plane of rotation. Without loss of generality, we define this to be horizontally polarised, denoted H. The orthogonal polarisation is then denoted V for vertical.

For this calibration measurement, we rotate the polarisation to be at 45°, such that the incoming beam can be described as an equal sum of components in H and V:

$$E_\mathrm{in} = E_V e^{i\phi_V} + E_H e^{i\phi_H}. \tag{2.17}$$

The phase profile of the H component is modulated by $e^{i\phi_{\mathrm{GL}}(x,y)}$ depending on the

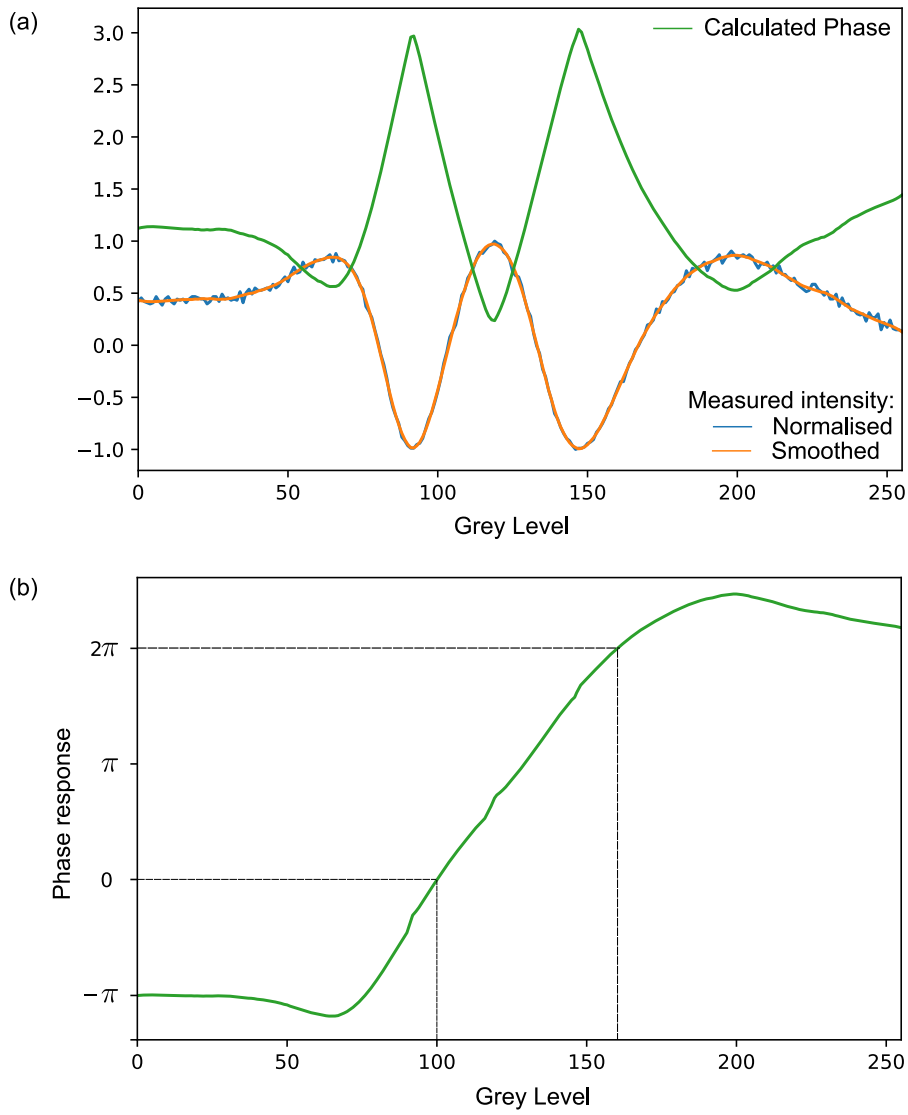


Figure 2.11: Example measurement and calibration curves to find the LC-SLM phase response as a function of pixel grey-level. (a) Intensity measured as a function of one pixel grey-level during calibration procedure using 45°-polarised light, normalised between ± 1 (blue curve) and smoothed (orange curve). Taking the inverse cosine gives the phase delay as a function of grey-level (green curve). (b) A monotonic function mapping phase-response to grey-level is generated for the LC-SLM pixel, by 'unwrapping' the phase curve above.

GL of each pixel, whilst the V component is unchanged. Therefore measurement of the output field after reflection from the SLM gives the intensity of the coherent sum

$$\begin{aligned} I_{\text{out}}(x,y) &= \left| E_V e^{i\phi_V} + E_H e^{i(\phi_H + \phi_{\text{GL}}(x,y))} \right|^2 \\ &= |E_V|^2 + |E_H|^2 + 2E_V E_H \cos\left(\phi_V - \phi_H - \phi_{\text{GL}}(x,y)\right). \end{aligned} \tag{2.18}$$

We can therefore extract the phase delay $\phi_{\text{GL}}(x,y)$ as a function of pixel GL, by

measuring the output intensity of each pixel as the GL is continuously scanned through all 256 values. In practice, a camera image of the entire 2D output plane is captured as each GL value is uniformly applied to all pixels simultaneously, and this image is then processed to map the measured intensity profile to the SLM pixel array. Fig 2.11(a) shows the measured intensity for a single example pixel as a function of applied GL. The curve is normalised by removing the offset and fixing maximum and minimum values to ± 1, and a filter is applied to smooth the curve. The phase delay ϕ_{GL} is then calculated by taking the inverse cosine of this smoothed curve. This phase curve can be 'unwrapped' to produce a monotonic function of phase response as a function of GL, which is plotted in Fig. 2.11(b) for the example pixel. This process is repeated across all SLM pixels, to produce a similar GL-Phase response curve, and a suitable range of GL values that provide a 2π phase delay for all pixels was identified. For the pixel shown, the GL values ranged between 100 and 160.

Using these response curves, a custom LUT is created, which is used to convert a given phase profile to the required GL values. A new LUT is created for all the LC-SLM models used throughout this work.

2.3.2 Curvature of the DMD and LC-SLM

Due to imperfect fabrication processes, the back panels of both the DMD and LC-SLM have significant curvature, on the order of tens of wavelengths. Therefore after reflection from either SLM, the phase profile of the beam is severely distorted. We must compensate for this wavefront curvature, and in fact the LC-SLM can be used to make the correction, since our encoding technique allows us to arbitrarily modulate the beam phase profile with the LC-SLM.

To precisely measure the wavefront distortion created by each SLM, we perform an interferometric measurement using the SLM and a uniform plane mirror. We display on the SLM a uniform amplitude and phase pattern, however the curvature of the SLM will modulate the beam as

$$E_{\mathrm{SLM}} = e^{i\phi_{\mathrm{curv}}(x,y)}. \tag{2.19}$$

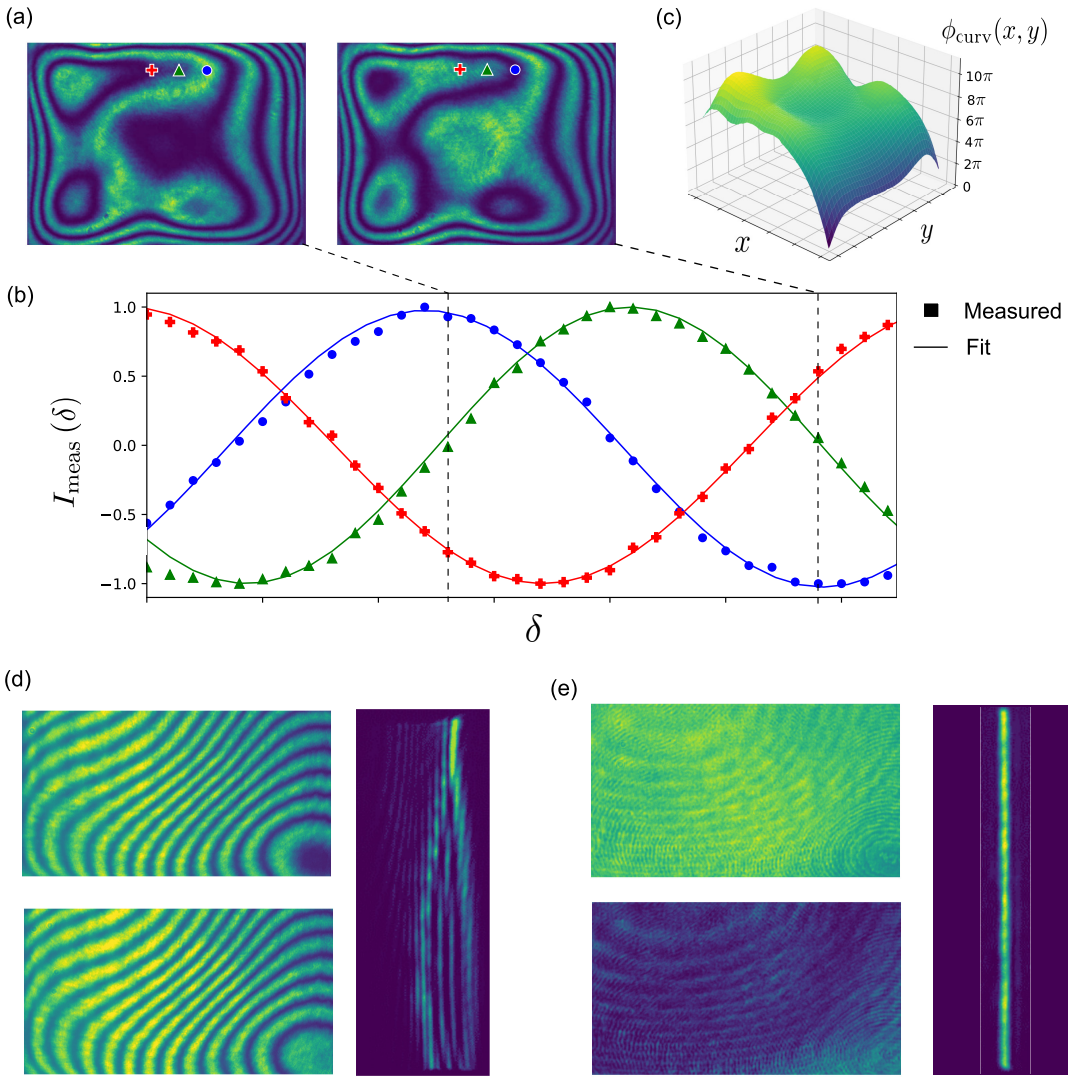


Figure 2.12: Calibration of the wavefront distortion caused by LC-SLM and DMD backplane curvature. (a) Images from experiment of the interference pattern generated after reflection from the DMD. (b) Intensity measurement as a function of reference delay length, for pixels located approximately shown in the images above, and the numerically fit cosine curve. (c) Wavefront curvature profile extracted from the cosine fitting parameters of each pixel. (d) Interference pattern generated after reflection from both the DMD and LC-SLM, and the field measured at the output vector plane. (e) The same images, taken after the wavefront curvature correction has been applied.

We use a piezo actuator to vary the path length difference, δ, between the SLM and reference mirror, and the measured intensity at the output plane is

$$I_{\text{meas}}(x, y, \delta) \propto \left| e^{i\phi_{\text{curv}}(x,y)} + e^{i\phi(\delta)} \right|^2 \tag{2.20}$$

$$\propto 2 + 2\cos\left(\phi_{\text{curv}}(x, y) - \phi(\delta)\right), \tag{2.21}$$

We record the full 2D interference pattern as the delay length is modulated, then resize all the images to match the SLM pixel dimension. Each pixel is independently renormalised to remove the offset and set maximum and minimum values at ± 1. Finally, the renormalised data is numerically fit to a cosine function with an appropriate period, such that the only fitting parameter is the phase offset. The value of this fitting parameter gives $\phi_{\text{curv}}(x, y)$ for every pixel.

The images in Fig. 2.12(a) are experiment images showing the interference pattern measured after reflection from one DMD model, at two values of δ that give approximately a π phase delay. Many interference fringes are clearly visible, indicating a large curvature. Fig. 2.12(b) plots the renormalised intensity values recorded for three example pixels as δ is scanned, as well as the numerically fit cosine curves. The position of each pixel is indicated in (a) by the respective symbols. The final calculated wavefront curvature profile $\phi_{\text{curv}}(x, y)$ is shown in Fig. 2.12(c).

Figure 2.12(d) shows images of the interference pattern between the combined DMD and LC-SLM used in later experiments, and a uniform reference beam, with a π phase delay applied between top and bottom photos. Also shown is the MVM output plane after the cylindrical lenses, with the output clearly distorted.

Figure 2.12(e) shows the same images, after a wavefront correction has been applied with the LC-SLM; when encoding the weight matrix pattern with the phase grating technique, (2.13) is modified to compensate for the curvature of the particular DMD and LC-SLM models being used, becoming:

$$F(x, y) = \phi(x, y) - \phi_{\text{curv}}^{(\text{DMD})} - \phi_{\text{curv}}^{(\text{LC-SLM})} + \pi \left(1 - M(x, y)\right). \tag{2.22}$$

With this correction applied, the interference measurement was retaken at both the LC-SLM and output vector planes. The fringes have been almost entirely removed, and the output field is now a narrow straight line, as expected. The few low-contrast artifacts that are visible are attributed to 2π jumps in the LC-SLM phase grating changing the signal amplitude, and a cavity effect from the cover glass of the camera, and not from distortion of the wavefront.

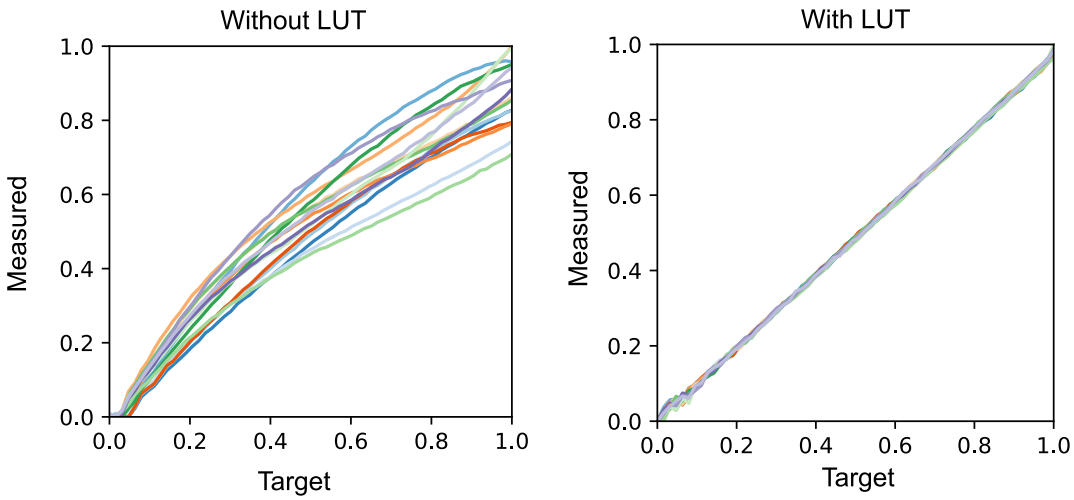


Figure 2.13: Normalised amplitude of each matrix element generated by the LC-SLM, measured as a function of the target input value, before calibration (left panel) and after the calibration LUT is applied (right panel).

2.3.3 Precise MVM amplitude correction

To get accurate MVM results, we need to conduct a precise calibration procedure, using measurements from the output of the optical multiplier itself. The following calibration procedure was regularly repeated, especially when the optical path or the geometry of the patterns displayed on the DMD and LC-SLM were changed.

Recall we want to display a target matrix W on the LC-SLM of dimension $N \times M$, indexed with i and j, and measure an output vector v of dimension M, indexed with j. We generate a phase grating to encode this matrix using target fields $A(x, y) \propto |W_{ij}|$ and $\phi(x, y) \propto \arg(W_{ij})$.

If we generate $A(x, y)$ and $\phi(x, y)$ naively from W, we find the output vector responds non-uniformly and non-linearly to each matrix element. This error can be attributed to several sources. First, the system is illuminated with a broadened Gaussian beam, meaning elements at the edges will be inherently dimmer than those at the centre. Second, optical aberration from non-ideal components, such as the cylindrical lenses, will distort the beam as it propagates through the system. Finally, the LC-SLM encoding technique is not perfect: higher diffraction orders can 'leak' through the spatial filter, the pixelated nature of the SLM reduces the blazed grating efficiency, and the previously-described calibration efforts can not

perfectly correct the voltage-phase response and wavefront curvature. Since all these error sources are systematic, we can calibrate our system to correct for them. Here we describe the amplitude calibration, the procedure that helps us to correctly generate the target field $A(x, y)$ given a target matrix W.

We use the DMD to illuminate just one matrix element W_{ij}, scan its value from minimum to maximum in 64 uniform intervals, and record the measured amplitude of the j^{th} output vector element. We repeat this for all $N \cdot M$ elements, and normalise all the curves by the largest measured amplitude. Figure 2.13(a) plots the 15 normalised curves measured for an example matrix of size 3×5. The non-uniform and non-linear response of each element is evident. By inverting these curves, a LUT can be created, mapping the desired output amplitude of each matrix element to the target amplitude that should be encoded in the phase grating technique.

To verify the calibration is working, the procedure of individually scanning each element is repeated, this time using the LUT. Once again all 15 curves are globally normalised by the maximum amplitude. The result is plotted in Fig. 2.13(b), with all elements now showing excellent uniformity and linearity.

2.4 Conclusion

In this chapter, we have examined the implementation of optical matrix-vector multiplication using free-space optics and spatial light modulators, including the crucial elements of optical fan-out and fan-in.

As motivated in Chapter 1, in order for MVM-based ONNs to eventually outperform traditional digital processors, it will be necessary to significantly increase the scale of matrix size implemented in a single optical MVM. So far in our experiments we have demonstrated a maximum matrix size of 56×56, and in the next chapter we will show how this was further improved to 200×50. The pixel resolution of modern SLMs can be as large as 2000×1000, so at least in principle the matrix size can be readily increased by another order of magnitude. At this scale it may already be possible to demonstrate performance at least

matching digital electronics, depending on the energy efficiency of analog-to-digital conversion of the 1000 vector elements.

Another crucial element for the success of optical computing will be achievable arithmetic precision. Many applications in high performance computing require extremely high precision arithmetic, and it is common for modern digital processors to support up to 64-bit floating-point numbers. However neural networks are an exception, and it is common for models to use far lower precision, with 8-bit integer arithmetic being common.

The precision with which the vector and matrix elements are encoded is dependent on the devices being used, and the requirements of the application. Each pixel of an LC-SLM independently allows multi-bit control, with the particular models used in our experiment allowing eight bit modulation. In contrast, our multi-level encoding scheme using the binary DMD pixels demonstrated a trade-off between achievable bit-depth and maximum dimension. In the next chapter we opt to use the largest possible dimension, and therefore use vector encodings as low as two bits. In contrast, in later chapters we use much smaller matrix dimensions, allowing us to use far higher bit-depth encodings.

However, unlike digital processors the overall arithmetic precision of an analog MVM is not solely dependent on the vector and matrix encoding bit-depths. For our optical multiplier, any errors in the physical system may reduce the multiplier precision. We therefore detailed some of the calibration procedures employed to minimise these system errors. The important figure of merit is the signal-noise ratio of the complete MVM, and our experiments in this chapter demonstrated a large SNR is possible that is broadly independent of matrix dimension (within the limits of the devices used.) In the next chapter we show the SNR achieved is sufficient to successfully deploy our optical multiplier as part of an ONN, and the ability to maintain the SNR whilst increasing matrix dimension will be vital to scaling optical multipliers for larger and deeper ONNs.

3

Hybrid training of optical neural networks

Contents

Having demonstrated a reconfigurable optical multiplier that utilises real-valued matrix elements, we looked to improve the system in multiple ways. First, we significantly increased the matrix size, achieving results up to size 200×50 without compromising the multiplier precision. Next, we improved the coherent detection method, removing the need for interferometric measurement. Finally, we encoded complex-valued matrix elements with the LC-SLM and performed complex-valued MVM.

With these advanced capabilities, we demonstrated how the multiplier can

be used as one of the layers in an ONN, and for the first time we demonstrate how the optical multiplier can be used in every iteration of network training, a process we refer to as *hybrid training*. Demonstrating this hybrid training scheme is an important intermediate step towards the larger goal of performing *optical backpropagation*, when all steps in the neural network training, including multiple linear and non-linear layers, are performed using optics.

Many of the methods and results in this chapter are presented in the article *'Hybrid Training of Optical Neural Networks'* [137] (Author contribution: joint-first author, carried out experiments and performed data analysis. Equal contribution to manuscript preparation.)

3.1 Improved Optical Multiplier

3.1.1 Larger matrix dimension

To improve our optical multiplier, the system was rebuilt with new models of both the DMD and LC-SLM, higher quality cylindrical lenses, and a new optical path design allowing higher resolution camera imaging. The optical system still followed exactly the concept outlined in the previous chapter.

The new LC-SLM used was a Santec SLM-100, with pixel resolution 1440×1050 and pixel pitch 10.4 μm. This model had the additional benefit of 10-bit voltage control, meaning the phase response was controlled by 1024 grey-levels values, giving far more precise control. The DMD model was a DLP Light Crafter 6500EVM, with pixel resolution 1920×1080 and pixel pitch 7.56 μm. The refresh rate of this model is 10 kHz, meaning the input vectors could be rapidly updated.

Using the upgraded optical multiplier, an MVM with matrix dimension 200×50 was achieved. As before, the input vectors were chosen randomly in the range $[0, 1]$, and matrix elements in the range $[-1, 1]$. For this measurement, we did not perform coherent detection and instead just used the intensity measurement to find the absolute value of the output. One example MVM result plotting the $M = 50$ measured and theory output values is shown in Fig. 3.1(a), and the results for 50 random MVMs are presented in Fig. 3.1(b). Despite the increase

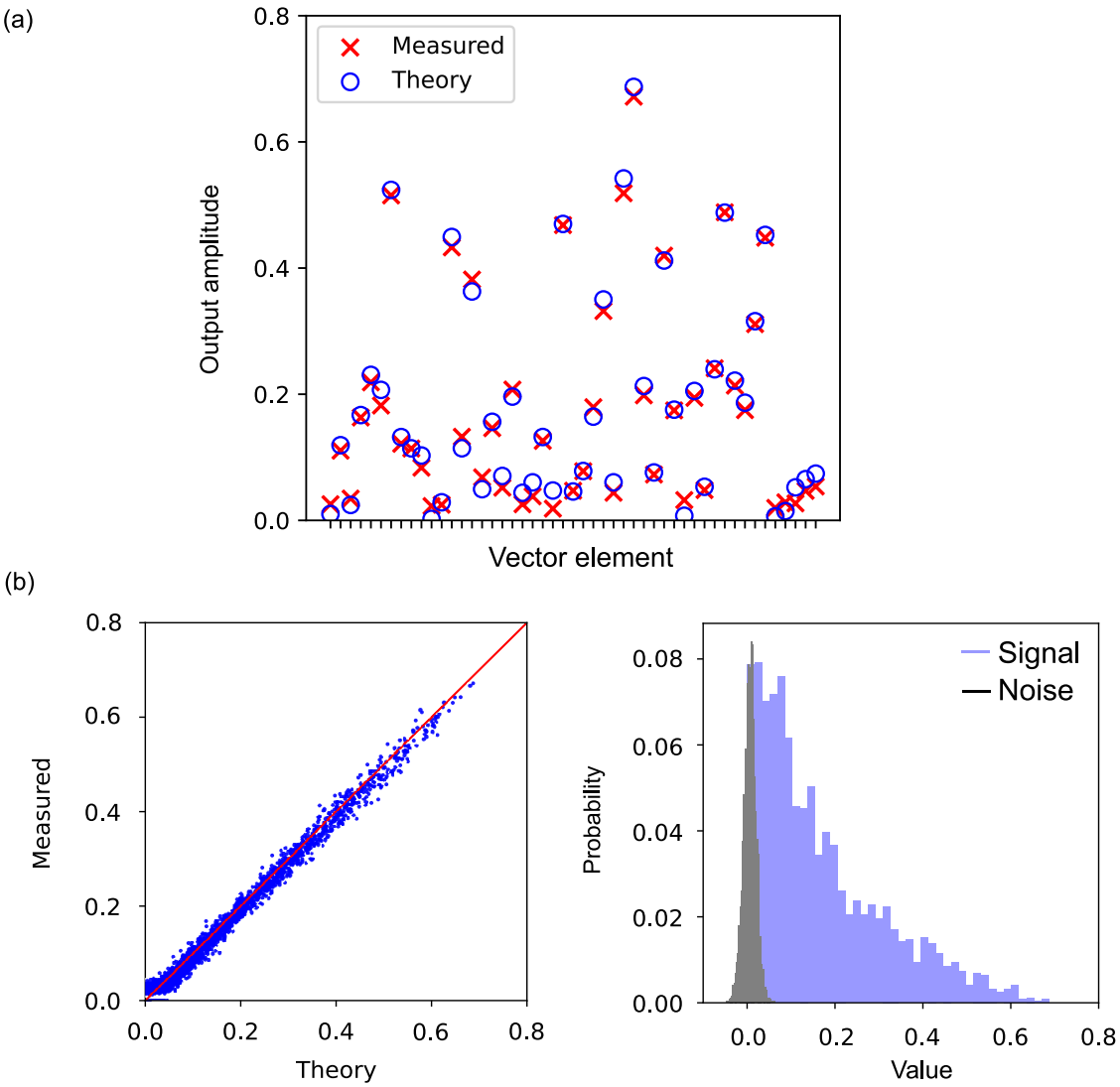


Figure 3.1: Optical MVM results for a matrix dimension of 200 × 50. Intensity measurements were used to calculate amplitude only. (a) Results for one example MVM. (b) Scatter of measured against theory values for 50 example MVMs, and the associated histogram of signal and noise. Comparing their widths gives an SNR of 10.2.

in matrix dimension, the SNR (as defined previously) was measured to be 10.2, consistent with our previous results.

When demonstrating our multiplier as part of an optical neural network, instead of encoding a matrix at this maximum size, we use a matrix of half the dimension, 100 × 25. This allowed us to double the grating period to 10 pixels, improving the reflection efficiency of the LC-SLM, further improving the SNR.

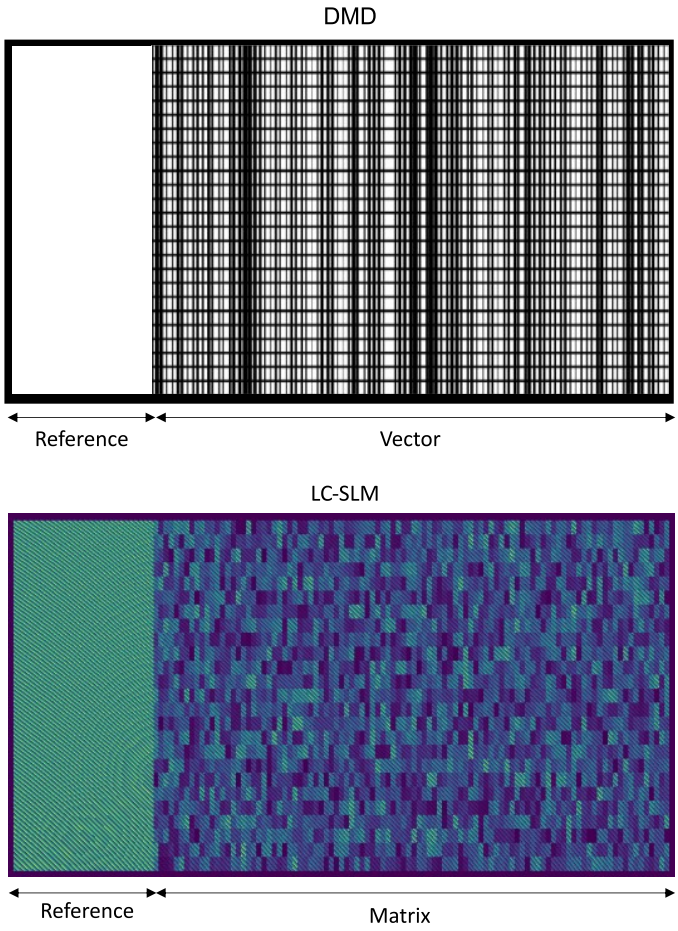


Figure 3.2: Example patterns displayed on the DMD and LC-SLM to perform optical MVM with matrix dimension 100×25, with single-shot coherent detection. The top image shows the binary DMD pattern, consisting of uniform reference region, and the 'fanned-out' vector. The bottom image shows the LC-SLM diagonal blazed grating pattern. The reference region is at maximum amplitude and maps precisely to the equivalent region on the DMD. Each matrix element is encoded with multiple grating periods for optimal SNR.

3.1.2 Single-shot coherent detection

The previous method of coherent detection required a series of many intensity measurements to be taken for each MVM, as the path length of the interferometer reference arm was scanned. However for real-valued MVM we do not need to measure an arbitrary phase value, as the output should only have phase values of 0 or π. It is possible to directly measure the real-valued output with a single-shot measurement, which allows much faster repetition rate of the MVM.

The output signal $E_{\text{sig}} = \pm |E_{\text{sig}}|$ is interfered with a uniform reference field

$E_{\text{ref}} = |E_{\text{ref}}| e^{i\phi_{\text{ref}}}$ with two strict requirements:

- The reference is phase-stable with respect to the signal: $\phi_{\text{ref}} = 0$.
- The reference is always brighter than the signal: $|E_{\text{ref}}| > |E_{\text{sig}}|$.

In this case, intensity measurement of the two fields yields

$$I_{\text{meas}} = |E_{\text{sig}} + E_{\text{ref}}|^2 \tag{3.1}$$

$$= |E_{\text{sig}}|^2 + |E_{\text{ref}}|^2 \pm 2|E_{\text{ref}} E_{\text{sig}}|. \tag{3.2}$$

We have used the knowledge that the signal phase only takes values of 0 or π, plus the requirements given above, to simplify the usual cosine term. We therefore find

$$I_{\text{meas}} = \left(E_{\text{ref}} \pm |E_{\text{sig}}|\right)^2, \tag{3.3}$$

and we can extract the signed output field as simply

$$\pm|E_{\text{sig}}| = \sqrt{I_{\text{meas}}} - E_{\text{ref}}. \tag{3.4}$$

Therefore once E_{ref} is known, every MVM result can be found using a single camera intensity image.

When generated independently, keeping the signal and reference fields in-phase is difficult, and would require some form of active phase-locking. Any changes in path length difference, due to mechanical instability, vibrations and so on, can cause large phase fluctuations, which in turn introduces significant noise in the MVM precision. Phase locking the two arms of the interferometer used in the previous setup was investigated, using a piezo-driven mirror and PID control, however this method was not pursued. Instead a far more reliable scheme was devised: we generate the reference field using a dedicated region of the DMD and LC-SLM, and propagate the two fields simultaneously through the optical setup. Figure 3.2 shows example patterns displayed on the DMD and LC-SLM to generate the reference and signal. The DMD encodes a vector of size $N = 100$, the LC-SLM encodes a random matrix of size 100×25, and both are used to generate the reference field with the large uniform regions on the left.

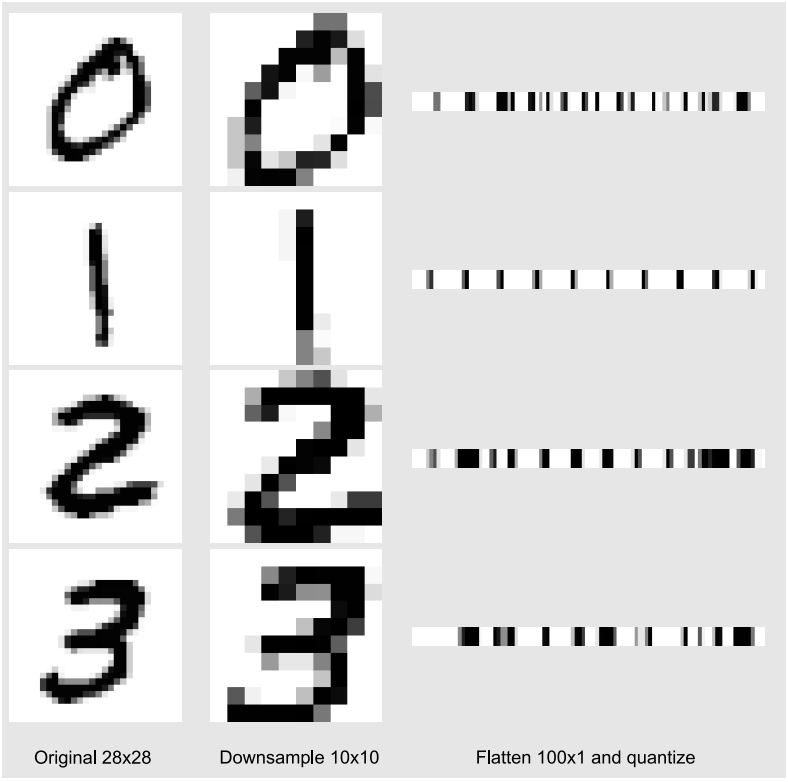


Figure 3.3: Examples of the preprocessing of the MNIST dataset performed for use with our MVM. Each image is downsampled, flattened and quantized, in order to be properly encoded by the DMD, with a maximum vector dimension of $N = 100$ and 4-bit precision.

The cylindrical lenses that perform the 'fan-in' process, now also act to combine the reference and signal beams, such that they interfere at the output plane. The reference and signal remain perfectly phase-stable, since the beam reflecting from both regions follows exactly the same path. Additionally, the LC-SLM can be used to conveniently set the precise phase difference. To ensure the reference intensity is always greater than the MVM output, the reference region must have sufficient width. In the extreme case, if all vector and matrix elements were equal to one, the reference region must be as wide as the signal region. However when used in practice, the MVM output values have a narrow distribution around zero, as explained below, and a smaller reference region is sufficient. In Fig. 3.2 the reference region has a width of 300 pixels, whilst the signal region is 1140 pixels wide.

The reference region essentially acts as an additional set of matrix columns that always contain the value one, such that the MVM output is always positive, by adding a suitably large bias. In this case no information is lost by performing

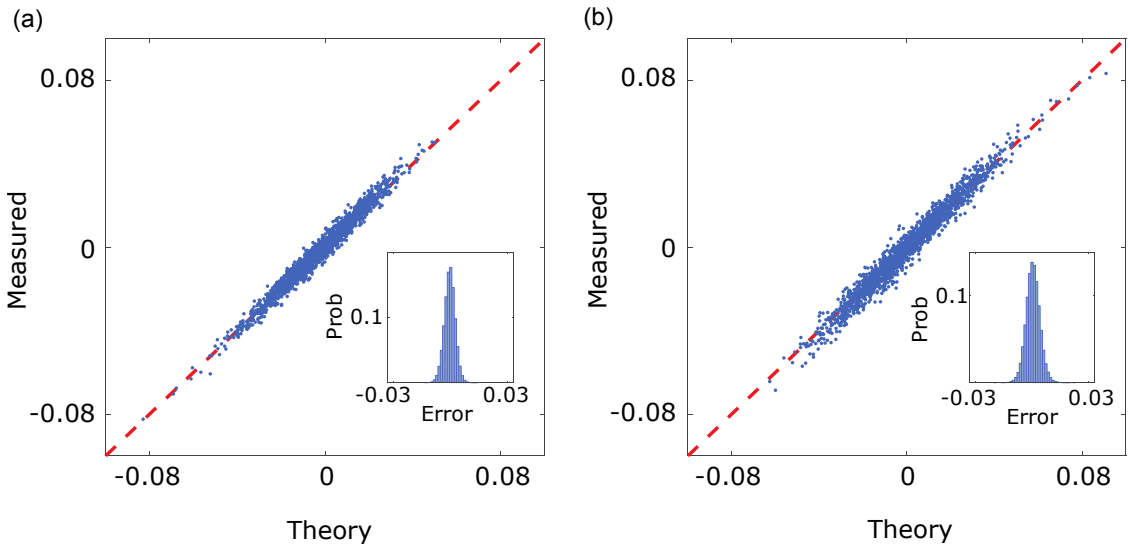


Figure 3.4: Scatter plots of measured against theory values for real-valued MVM at larger matrix sizes and improved precision. Vector inputs are preprocessed MNIST images, and the weight matrices are random matrices of size (a) 100×10 and (b) 100×25. Histograms of the error between measurement and theory are shown inset. Compared to our previous MVM results, the error RMSE is an order of magnitude lower. Figure adapted with permission from [137] © Optica Publishing Group.

intensity measurement, and subtracting the bias gives the final real-valued result. In practice, the exact value of the bias does not matter. Instead, the intensity output measured by the camera for each output element is independently mapped to the target value by simple least-square fitting over many examples.

3.1.3 MNIST dataset and improved multiplier precision

The random vectors and matrices we were previously using to characterise our MVM were chosen to give a broad range of output values, and our emphasis was on maximising the SNR over this range. However, when applied in a neural network, it was empirically found the MVM output values had a much narrower distribution around zero. It is critical our MVM has good SNR over this narrow range, so we characterise the MVM using the matrices and inputs typical of those we will use in the neural network training.

The dataset we use throughout this experiment is the one most commonly used to demonstrate and benchmark machine learning algorithms: the MNIST (Modified National Institute of Standards and Technology) database of handwritten

digits [158]. It contains 70,000 images of the digits 0-9 drawn by hand, and the task is to classify the images into the 10 classes. Each image is 28×28 pixels, with grayscale values. For our network, we downsample the image to 10×10, flatten to a vector of dimension $N = 100$, and quantize the graylevels to 4-bit precision, to match the DMD precision of 15 encoding levels (15 physical pixel width per logical pixel). This preprocessing is shown in Fig. 3.3.

We recharacterize our MVM precision by running a large number of real-valued MVMs, using a random sample of the preprocessed MNIST images as the vector inputs, and random matrices with elements drawn from a normal distribution with zero mean and standard deviation 0.5, denoted $N(0, 0.5)$. We use two different matrix sizes of 100×10 and 100×25, and the results are shown in Fig. 3.4(a) and (b) respectively. As before, the output vector is normalized by the maximum possible output which is obtained when all the vector and matrix elements are at maximum. The RMSE for the two matrix sizes are 0.0024 and 0.0036, and a histogram of these errors is shown inset for both. Compared to our previous result, where our 56×56 multiplier had an RMSE of 0.0237, this represents an order-of-magnitude improvement in our multiplier precision. The scatter plots appear visually to show worse precision than Fig. 2.8, however note the output range is now less than 0.08.

3.1.4 Complex-valued MVM

We have demonstrated how using a coherent beam allows us to perform real-valued MVM, by applying a π phase shift for negative values. However, our LC-SLM encoding technique allows us to encode arbitrary phase values, not just 0 or π. Therefore our optical multiplier also naturally supports complex-valued operations.

Whilst the DMD still encodes positive-only values, the LC-SLM is used to encode arbitrary complex values, with matrix size 100×10. The final output vector is therefore also complex-valued, and to measure the real and imaginary parts of the output vector, we apply a phase shift ϕ_{ref} to the reference beam using the LC-SLM reference region. The real parts are extracted by setting $\phi_{\text{ref}} = 0$ and $\phi_{\text{ref}} = \pi$ and subtracting the measured intensities, and the imaginary parts are

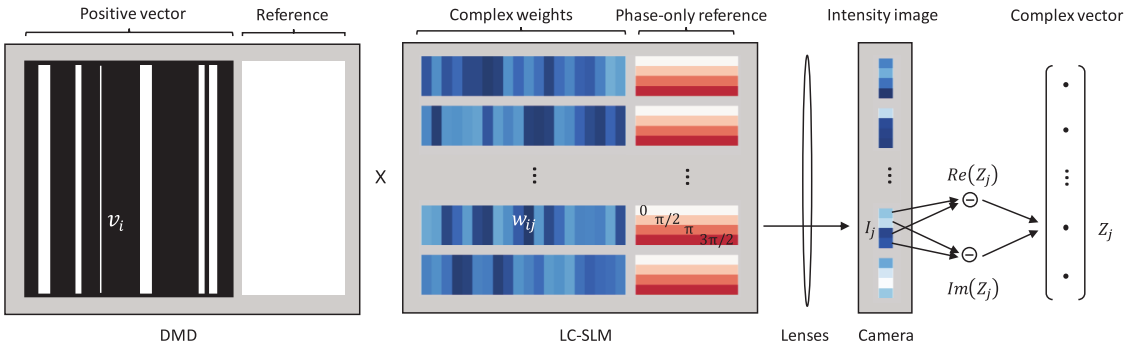


Figure 3.5: Conceptual scheme showing how to perform single-shot complex coherent detection. MVM is performed between positive-only vectors encoded by the DMD, and complex-valued weight matrix encoded by the LC-SLM. The phase-only reference on the LC-SLM splits each output element into four spots. Simple digital processing of the resulting intensity image yields the real and imaginary components of each vector element. Reprinted with permission from [137] © Optica Publishing Group.

extracted similarly by setting $\phi_{\text{ref}} = \pi/2$ and $\phi_{\text{ref}} = 3\pi/2$. Further details of the post-processing calculations are given in Appendix C.1. The calculated values are normalised as before, such that all matrix and vector elements at maximum gives an output of 1, for both real and imaginary components.

Instead of setting these four phases sequentially, and measuring the intensities with four separate camera images, we create four spatially-separated reference regions of different phases for each output element. That means each output is spatially split into four regions yielding the four required interference results. Therefore, the real and imaginary parts of the output can be read out from one single camera frame. This detection scheme is depicted conceptually in Fig. 3.5.

We show an image taken from experiment in Fig. 3.6(a), captured at the output plane for one example MVM with random complex-valued weights on the LC-SLM, and positive-only MNIST inputs on the DMD. We see the 10 output elements (indexed by $j \in [1, 10]$) are split into four distinct intensity spots corresponding to the four reference phase values. We plot the normalised intensity values extracted for the four regions of each output element in (b), and using these values we calculate the normalised real and imaginary components, and plot them on the complex plane in (c). We also plot the digitally-calculated theoretical value, and

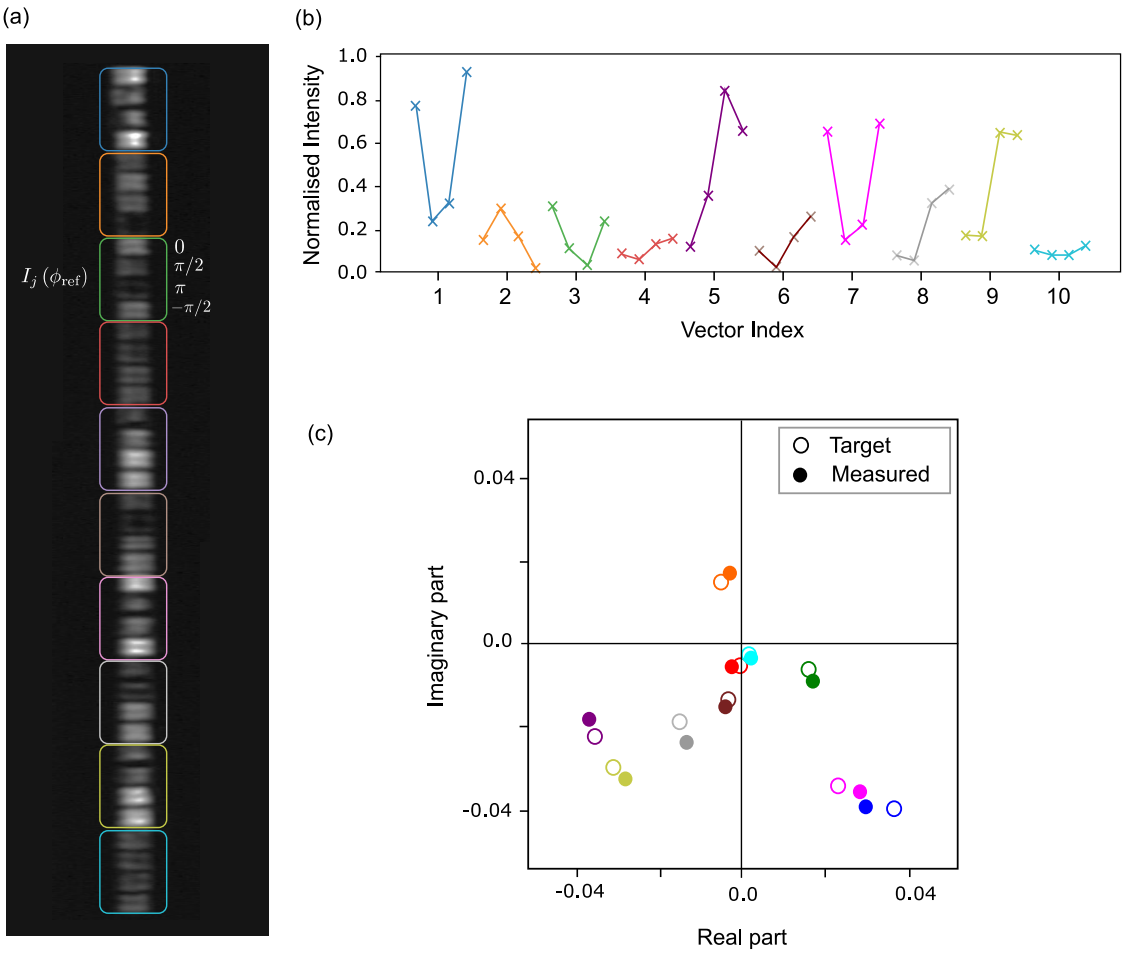


Figure 3.6: Example to demonstrate the complex-valued coherent detection method for one MVM with complex weight matrix of size 100×10. **(a)** The intensity image captured by the camera at the output plane, with 40 distinct spots visible. We highlight ten groups of four, corresponding to the ten output vector elements, each with a different reference phase values in $[0, \pi/2, \pi, -\pi/2]$. **(b)** The normalised measured intensity of the 40 spots, extracted from the camera image. **(c)** The measured and theoretical complex-valued output vector, plotted on the complex plane. For each element, the real component is calculated as the difference $I_j(0) - I_j(\pi)$, and similarly the imaginary component the difference $I_j(\pi/2) - I_j(-\pi/2)$. The theory value is the result of digitally calculated complex MVM.

once again we see good agreement between experiment and theory, although with some random noise for both components.

We repeat such a measurement for 100 examples with random MNIST inputs and random matrix elements - real and imaginary components drawn from a normal distribution $N(0, 0.5)$. For one output element ($j = 1$) we plot all the measured real and imaginary components (blue circles) on the complex plane in

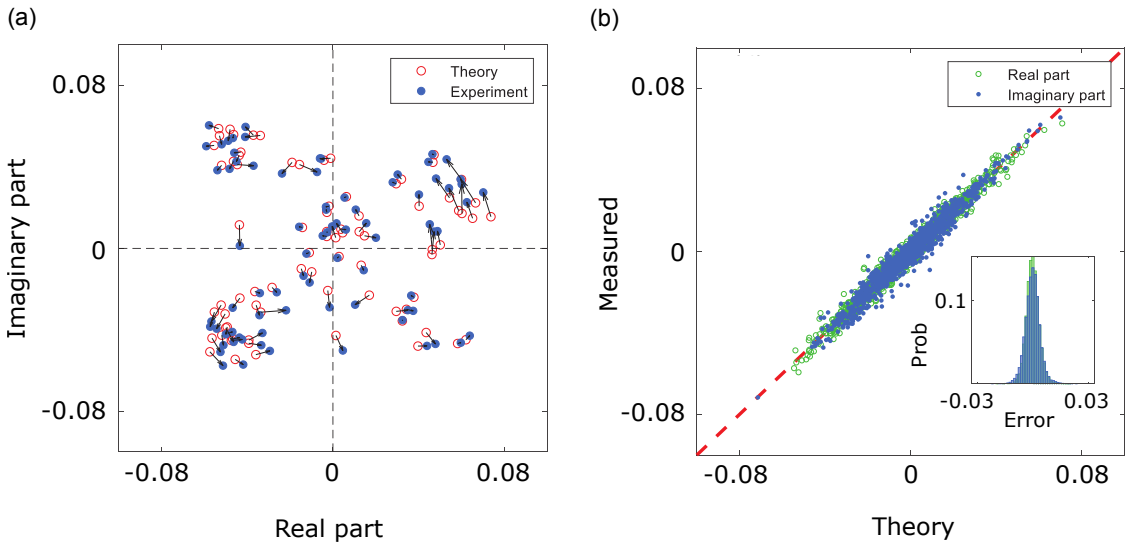


Figure 3.7: Results for 100 examples of complex-valued MVM with MNIST images as input vector and random weight matrix. **(a)** Real and imaginary components of the measured and theory values plotted on the complex plane, for the j=1 output vector element. Black arrows map associated points. **(b)** Scatter of measured against theory for all elements, for real and imaginary components. Error between measured and theory is shown inset. The RMSE is close to that for real-valued MVM. Figure adapted with permission from [137] © Optica Publishing Group.

Fig. 3.7(a). We also plot the digitally calculated theory values (red rings), and the errors between experiment and theory are indicated by black arrows. We see many of the measured values are rotated clockwise in the complex plane relative to theory, suggesting the primary error is a phase offset for this element. However in Fig 3.7(b) we separately plot scatters of the measured real and imaginary components (green and blue dots respectively) against theory values for all 10 output elements, and the errors distribute evenly around zero. The RMSE values are 0.0034 and 0.0039 for real and imaginary components respectively, only marginally larger than the real-valued MVM error.

3.2 Hybrid training

3.2.1 Concept

We construct various optical neural networks (ONNs), which use our physical optical multiplier to perform a portion of the computation. To train such an ONN, one can model the network architecture on the computer, and implement *in silico* (offline)

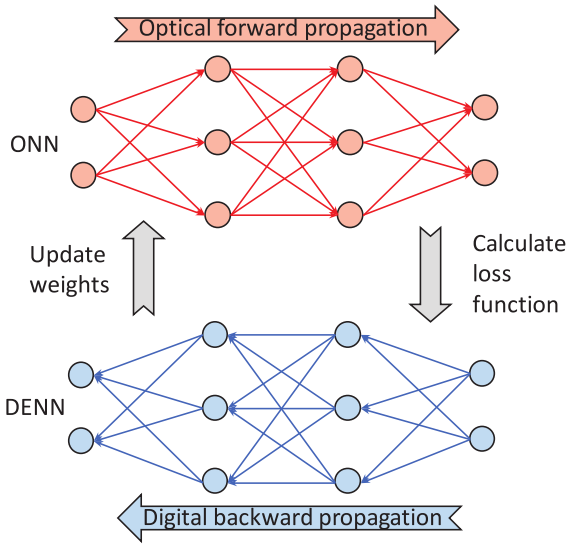


Figure 3.8: Concept of hybrid training of ONNs. Forward propagation in every training iteration is implemented in the optical system, whilst error backpropagation is implemented digitally. Figure adapted with permission from [137] © Optica Publishing Group.

training. The final weights after the training has converged are then transferred to the ONN to perform inference only.

However, as motivated in the introduction, this training scheme is flawed. The physical optical system contains aberrations that add noise to the multiplication result, as already explored in Ch. 2. It is extremely challenging to accurately model such aberrations with the digital system, and therefore the weights found by the digital model are likely not the optimal weights for the physical system. Instead, if the physical system can be incorporated in the training process, the aberrations and associated noises may be naturally accounted for, and weights can be found that give better performance on the inference task. This is the basis for our ONN hybrid training scheme, and is an example of in-the-loop training.

Recall that throughout this work we take training to mean the process of supervised learning, whereby the parameters of a neural network are iteratively updated via gradient descent, using a labelled dataset $(\mathbf{x}, \mathbf{t})$, where $\mathbf{x}$ are example network inputs, and $\mathbf{t}$ is the associated known label, giving the ground truth answer. Furthermore, we found that in each training iteration we calculate the

weight updates using the equation

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^{(l)}} = \sum_k \frac{\partial \mathcal{L}}{\partial z_k^{(l)}} \frac{\partial z_k^{(l)}}{\partial W_{ij}^{(l)}} \tag{3.5}$$

$$= \delta_j^{(l)} a_i^{(l-1)}, \tag{3.6}$$

needing the forward activations and backward errors at every layer. In our hybrid training scheme, the first layer activation vectors, $a_i^{(1)}$, are obtained from the output of our optical multiplier, after optical forward propagation. Then the error vectors, $\delta_j^{(l)}$, are obtained by performing backpropagation digitally. The two vectors are used to calculate the weight gradients, and the physical weights of the ONNs are updated accordingly. This process is then repeated across multiple mini-batches and epochs, with the physical system involved in every iteration.

3.2.2 Methods

	Description	Nonlinearity	Network size
ONN-1	Optical linear classifier	None	100×10
ONN-2	Hybrid opto-electronic network	ReLU	$100 \times 25 \times 10$
ONN-3	Complex-valued ONN	Modulus square	100×10

Table 3.1: List of ONNs demonstrated with our hybrid training scheme.

We demonstrate our hybrid training scheme with three different ONNs: an optical linear classifier, a more expressive network with optical and electronic layers, and a complex-valued ONN. The ONNs are summarised in Tab. 3.1, and each is described in detail in subsequent sections.

The ONNs are used to classify the MNIST dataset. The images are preprocessed to 4-bit vectors of size $N = 100$, as described in Section 3.1.3, and labelled by a binary 'one-hot' encoded vector of size 10, where all elements equal zero except the element indexed by the pictured digit. For example, all images of the digit zero are labelled by $\mathbf{t}^{(0)} = [1, 0, 0, \ldots, 0]$, images of digit one by $\mathbf{t}^{(1)} = [0, 1, 0, \ldots, 0]$ and so on.

The dataset is split into 60000 training images, 5000 validation images and 5000 test images. The training was done using 500 randomly-sampled mini-batches with a mini-batch size of 240, equivalent to two full epochs of the training set. Every

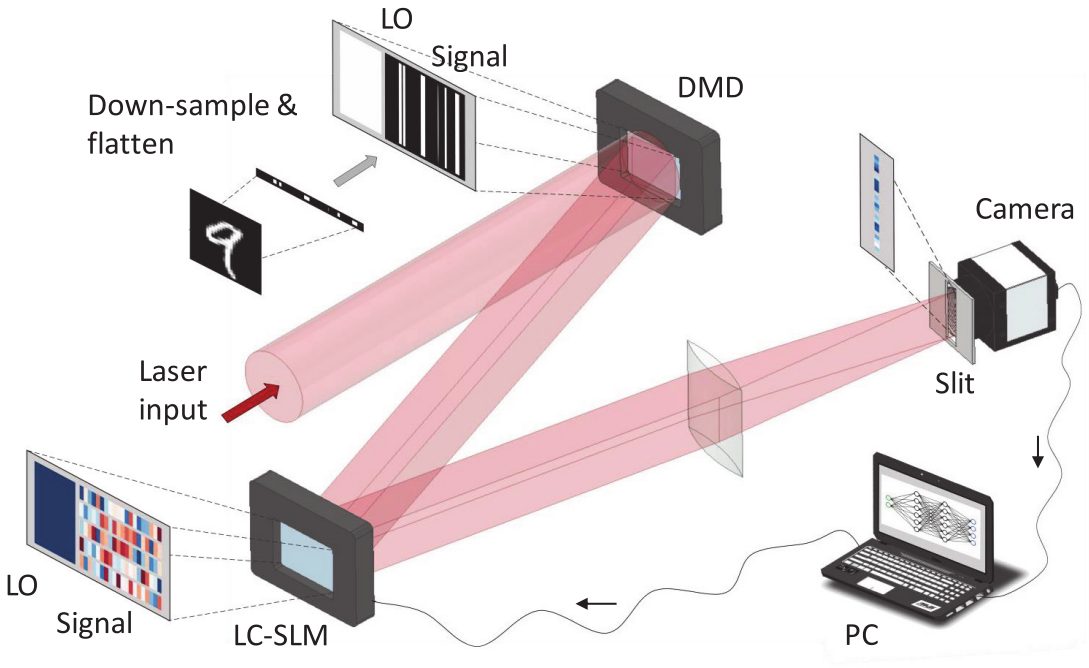


Figure 3.9: Simplified experiment scheme for hybrid training an ONN with one optical linear layer. Our optical multiplier serves as the basis for an ONN, utilising the properties of large scale, high SNR, arbitrary reconfigurability, single-shot coherent detection scheme, and use of real-valued or complex-valued weights. After detection with the camera, additional network layers can be implemented digitally, and backpropagation steps are also performed digitally. Figure reprinted with permission from [137] © Optica Publishing Group.

50 iterations, we calculate the 'validation accuracy' by performing inference on the validation set using the ONN. The test set is used to measure the final ONN classification accuracy after the entire training procedure has concluded.

The MNIST vector inputs are encoded with the DMD, and weight matrix is encoded with the LC-SLM, which both communicate with a PC using HDMI, operating at 60 Hz. At each clock cycle the DMD receives an RGB image (three 8-bit images), and internally each image is decoded into 24 individual binary bitmaps, displayed sequentially within each 60 Hz time slot. Therefore the DMD operates at an effective rate of 24×60 Hz $= 1.4$ kHz. In order to measure, transfer and process images at this rate, a camera with very high framerate (Basler ace acA640-750um) was used. The exact framerate was dependent on how many rows of pixels were used, ranging from 750 Hz to over 3 kHz, in the extreme case of using just a single

row of pixels. A hardware triggering system between the DMD and camera was created, to ensure the intensity image of the output field corresponded to the correct vector input. The LC-SLM has a much slower effective frame rate, limited by the liquid crystal response time (measured to be more than 200 ms) and rudimentary control software. The effective maximum frame rate was only 2 Hz.

The LC-SLM only needs to update once per mini-batch, therefore to minimise the overall computation time for our optical system, the batch-size should be very large, running as many frames as possible with the DMD at 1.4 kHz before each slow LC-SLM matrix update. In practice it was found 240 images could be reliably displayed at this rate without synchronisation issues. Therefore this was chosen as the mini-batch size.

In total, one training iteration took just less than a second, including 200 ms for the DMD to run one mini-batch and camera to transfer the measured images, 500 ms for the LC-SLM matrix update, and less than 300 ms overhead for digital calculations and control. Performing inference with the 5000 validation images was equivalent to 20 mini-batches, or approximately four seconds of run-time. So in total, with 500 training iterations, and 10 validation inferences, our hybrid training scheme took less than ten minutes to run.

3.2.3 Linear classifier and opto-electronic network

A neural network without nonlinear activation functions, with any number of layers, can be reduced to a single matrix transform, and is therefore equivalent to a single linear layer network. We call such a network a *linear classifier*. It will be limited to learning only linear mappings between inputs and outputs, so for many problems will have an upper bound of classification accuracy below an arbitrary neural network.

We construct an optical linear classifier (ONN-1), using our optical multiplier to directly connect the 100 input neurons, x_i, to the 10 output neurons, $z_j^{(1)}$. The real-valued weight matrix $w_{ij}^{(1)}$ therefore has dimension 100×10. The softmax function is digitally applied to the output vector measured by the camera, to yield the network output y_j. This architecture is shown in Fig. 3.10(a).

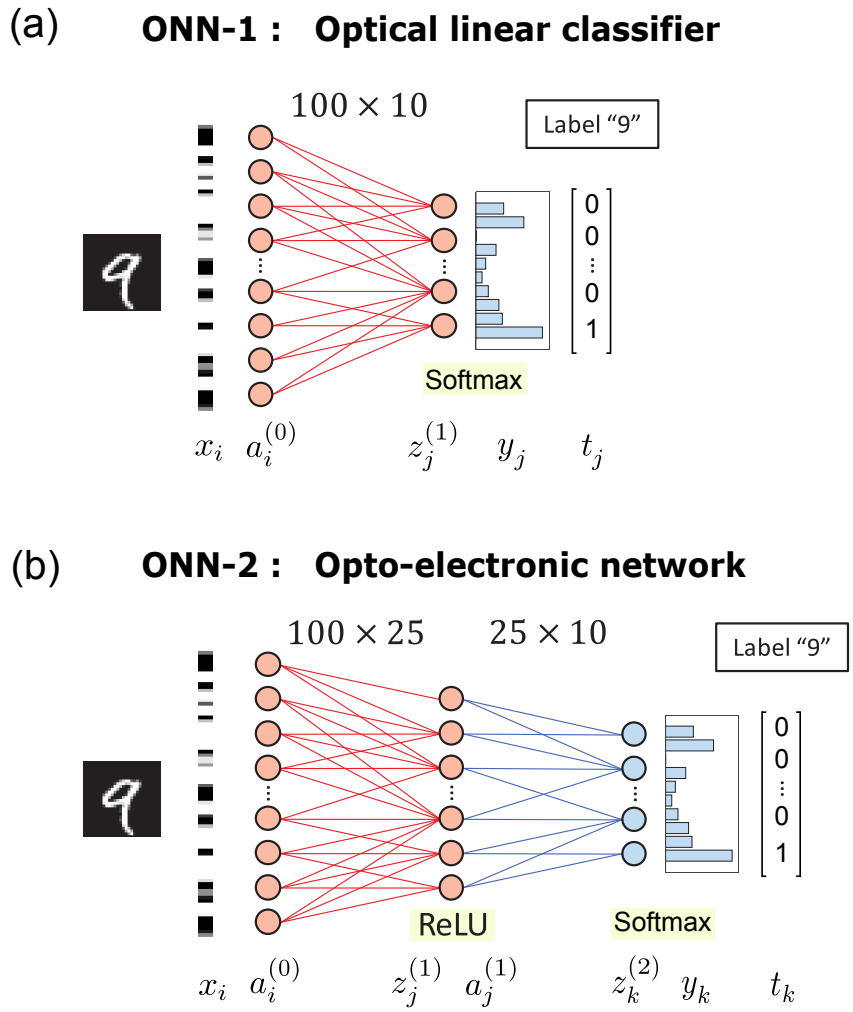


Figure 3.10: Network architectures of the real-valued ONNs used to demonstrate hybrid training. (a) Optical linear classifier (ONN-1) with 100 input neurons and 10 output neurons. (b) Hybrid opto-electronic network (ONN-2) with an optical layer, digital ReLU activation and a digital layer. The neuron numbers are 100, 25 and 10 for the input, hidden and output layer.

For this linear classifier, there is only one weight matrix to update, and the gradients are calculated as

$$\frac{\partial \mathcal{L}}{\partial w_{ij}^{(1)}} = \sum_k \frac{\partial \mathcal{L}}{\partial z_k^{(1)}} \frac{\partial z_k^{(1)}}{\partial w_{ij}^{(1)}} \tag{3.7}$$

$$= (y_j - t_j) \cdot x_i, \tag{3.8}$$

where y_j is calculated by optical forward propagation in every iteration. No MVM is required in the digital backpropagation step, and so the majority of computation in this hybrid training scheme is performed optically.

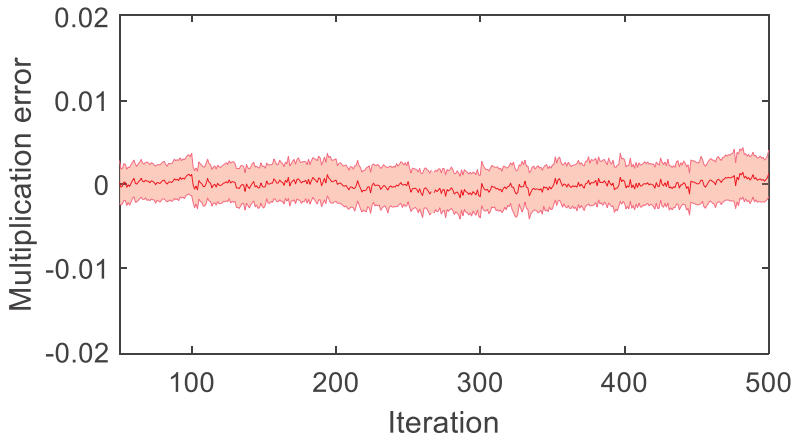


Figure 3.11: Evolution of measured noise in ONN-1 over the duration of hybrid training. The mean and one standard deviation of the error distribution for each training iteration is plotted. Reparametrisation after each epoch maintains the noise at a consistently low level. Figure adapted with permission from [137] © Optica Publishing Group.

Before the training starts, the weight matrix is initialised with a normal distribution $N(0, 0.5)$, intensity measurements are taken for one random mini-batch of input vectors, and straight-line fitting is used to determine the offset and scaling parameters between camera intensity measurements and the theoretical digitally-calculated values. It is important that the error in our MVM does not accumulate and blow up during the hybrid training, and to help mitigate this we reparametrise the straight-line fitting between camera intensity measurements and the theoretical values after every epoch. This was required as the parameters are found by performing a batch of MVMs with a single fixed weight matrix, which adds some unique systematic error to the measured parameters. Therefore as the weight matrix is updated and evolves during the training process, the systematic errors in the output distribution are slightly modified and must be corrected. In Fig. 3.11 we plot the evolution of the optical MVM error during the entire training process, i.e the difference between optically measured values and the theoretically expected results. It is clear that our optical system maintains excellent precision throughout. It is important to note this error is present only for the actual physical system. When performing the digital backpropagation, we do not need to model the optical system in any way, or consider any noise in the calculations.

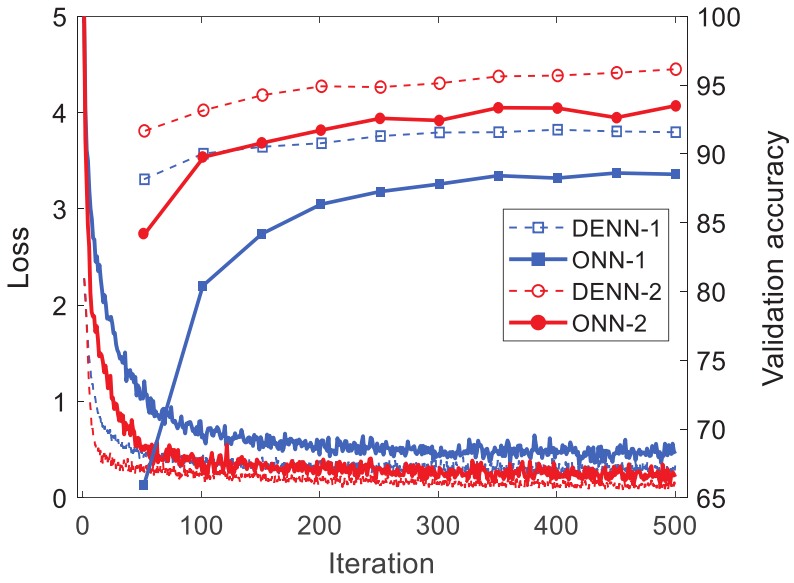


Figure 3.12: Learning curves for ONN-1 and ONN-2 during hybrid training, benchmarked against their digital equivalent networks. The loss is recorded after every training iteration, and validation accuracy after every 50 iterations. Adapted with permission from [137] © Optica Publishing Group.

The learning curves (value of the loss function and the validation accuracy) for ONN-1 are plotted in Fig. 3.12, and we see that both the loss function and validation accuracy converge quickly after the first few iterations. We compare how well our ONN performs compared to a similar digital electronic linear classifier (DENN-1). This network uses exactly the same architecture and hyperparameters, but is trained and tested digitally on a computer. The learning curves are also plotted in Fig. 3.12, and we see slightly faster convergence and higher classification accuracy compared to ONN-1. The reason for the slightly degraded performance of ONN-1 is random dynamic experimental noise, and this is explored in more detail in Section 3.2.6.

After we have trained our ONN, we perform image classification on the test dataset, using our optical classifier to perform inference. Fig. 3.13(a) shows an example of the intensity profile measured by the camera at the output plane of ONN-1. The top image shows the intensity of the reference field only. The intensity distribution is non-uniform due to the Gaussian profile of the laser beam. The bottom image shows the combination of reference and signal, when the input vector is an MNIST "4" and the matrix is the hybrid-trained weight matrix. The output values are digitally extracted from the areas marked in red rectangles, directly from

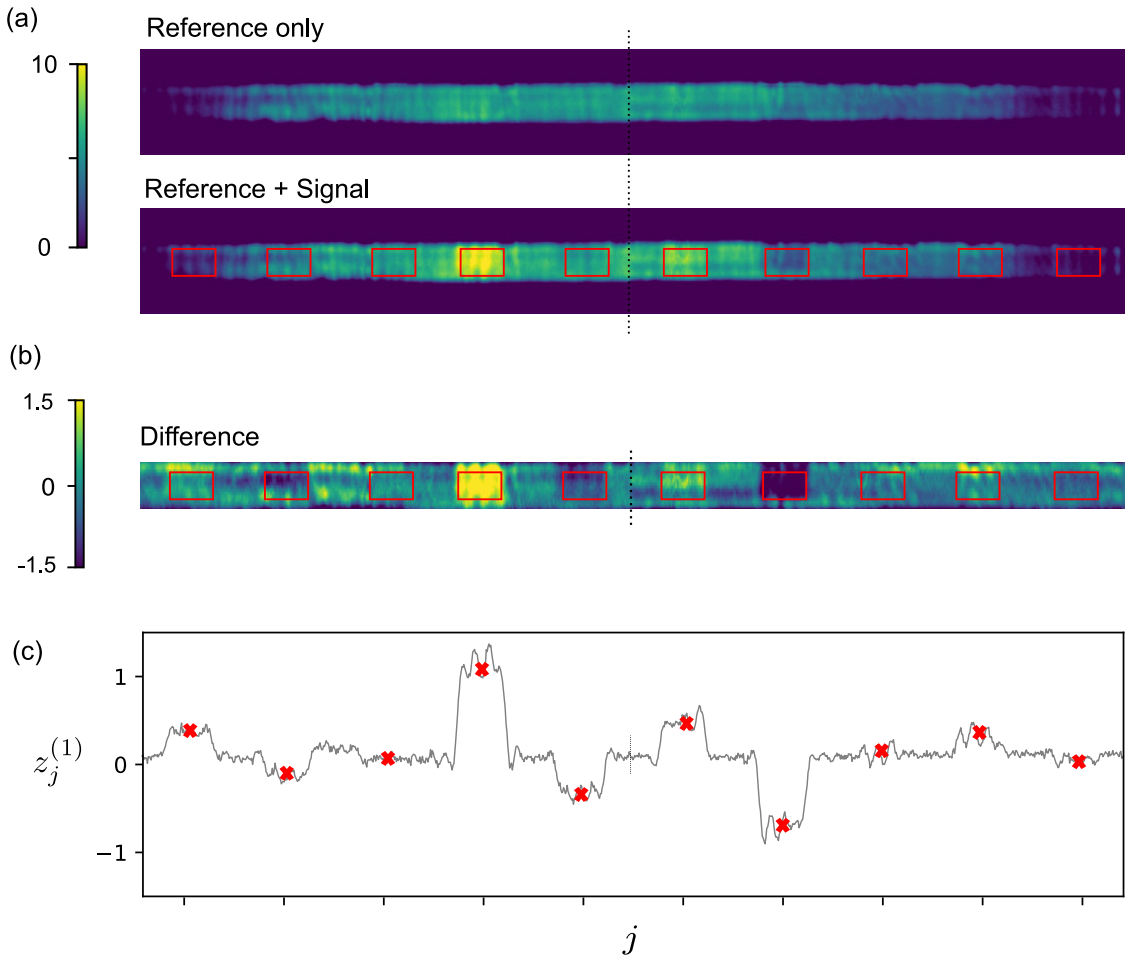


Figure 3.13: Example field intensity produced at the output of ONN-1 to classify an MNIST digit. In experiment to increase camera resolution, each half of the field was measured separately, and the images shown here are digitally reconstructed to show a single image. (a) The field intensity for reference beam only (top panel) and reference and signal (bottom panel), with an MINST "4" as input, and hybrid-trained weight matrix. (b) Difference between the reference-only and reference+signal images, found digitally in order to visualise the real-valued field. (c) Cross-section of the central part of the image, and the calculated intensity for each output vector element. The brightest peak at $j = 4$ shows the ONN has successfully classified the image.

the bottom image, however it is insightful to plot the difference between the two images, which reveals the real-valued field encoding the MVM result. This is shown in Fig. 3.13(b), and a cross-section of the central part of the image is shown in (c). The fourth output vector element is clearly the brightest, as expected.

Similar plots for successfully-classified examples of all ten digits are shown in Fig. 3.14. In each case the index of the brightest spot clearly corresponds to the input digit. Obviously not all of the 5000 test images are correctly identified, and

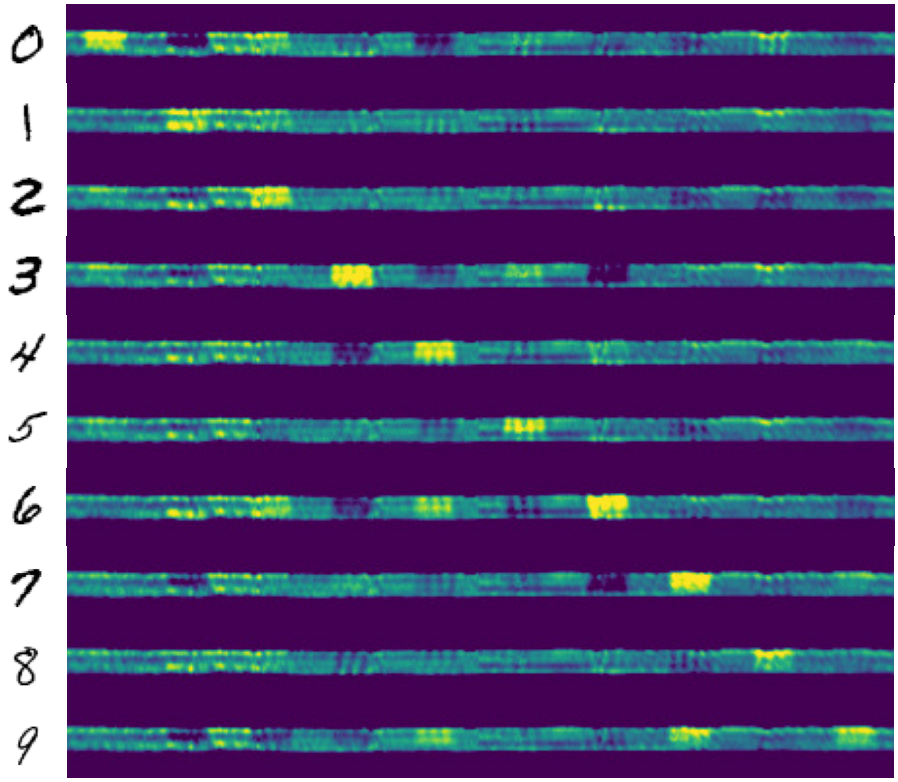


Figure 3.14: Example field intensities measured at the output of ONN-1 when each MNIST digit was successfully classified. In each case the brightest element index corresponds to the input digit.

we can visualise how well the network performs with a 'confusion matrix', shown for ONN-1 in Fig. 3.15(a). Taking all images with a given predicted label, the confusion matrix shows the percentage of those images with each true label (such that the columns add to 100%.) The overall classification accuracy for ONN-1 test set is 88.0%, and the confusion matrix shows this is well-balanced across all the labels. For comparison, DENN-1 test set scored 91.8%.

To improve the classification accuracy, we build a more complicated opto-electronic network (ONN-2) consisting of one optical layer and one digital layer. The first layer uses our optical multiplier to connect the 100 input neurons, x_i, to 25 hidden layer neurons, $z_j^{(1)}$. After detection, a digital ReLU activation is applied, and one digital output layer connects the 25 hidden layer activations $a_j^{(1)}$ to the 10 output neurons $z_k^{(2)}$, where the softmax function is applied to get the network outputs y_k. This ONN-2 architecture is shown in Fig. 3.10(b).

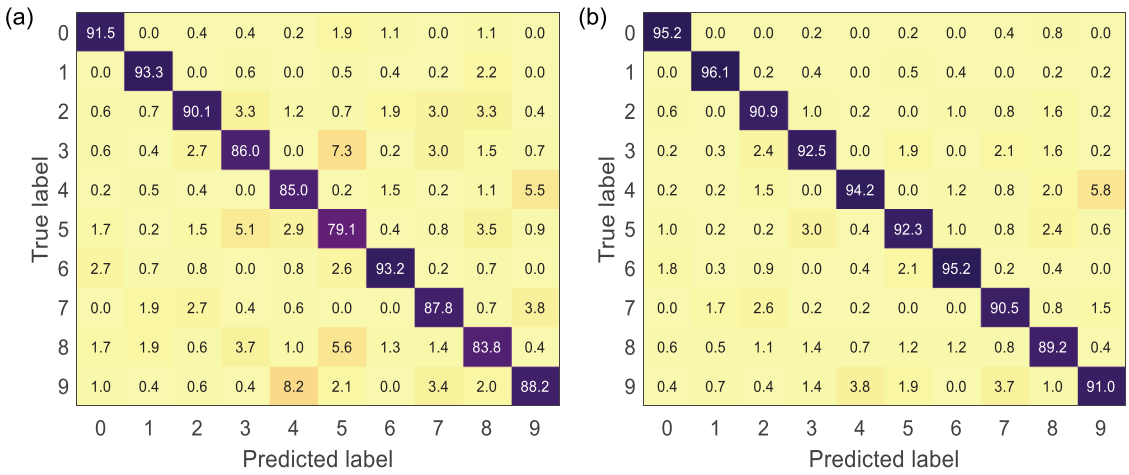


Figure 3.15: Confusion matrices for the test set of (a) ONN-1 and (b) ONN-2. Figure adapted with permission from [137] © Optica Publishing Group.

Hybrid training is performed as before, with the output of the optical multiplier being used in every training iteration to calculate the weight gradients. Fig. 3.12 also plots the learning curves of ONN-2, which reaches 92.7% accuracy for the final validation set, outperforming both ONN-1 and DENN-1. The confusion matrix, shown in Fig. 3.15(b), again shows the classification is well balanced amongst all classes. As before, we compare with an identical digital network DENN-2, which achieves slightly faster convergence and higher final accuracy of 95.7%.

3.2.4 Complex-valued ONN

As described in Section 3.1.4, our optical multiplier can perform complex-valued MVM. We can therefore construct an ONN that uses complex-valued weights, with each element encoded by a real and imaginary component. For a fixed layer size, we might naively expect the complex-valued network to achieve a higher classification accuracy, given it has double the number of tunable parameters. However if the network remains linear, additional parameters will not help overcome the linear classifier limit, which was empirically found to be 92.2% for this dataset.

In order for complex-valued networks to improve the classification accuracy, some form on nonlinearity must be introduced. It has been shown that diffractive neural networks that use complex-valued parameters outperform linear classifiers, even though the diffractive layers are still linear [57, 58]. The improved performance

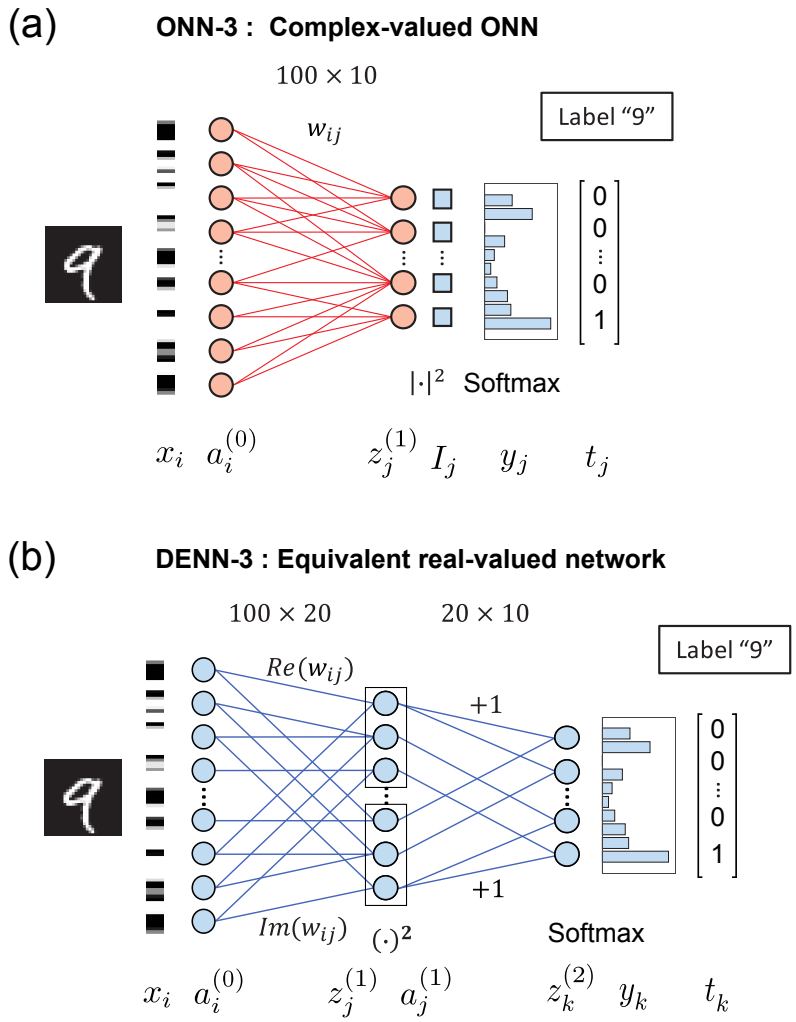


Figure 3.16: The network architecture of the complex-valued ONN-3 is equivalent to a real-valued network with additional hidden layer. (a) The single-layer complex ONN architecture. (b) Equivalent real-valued network, with an additional hidden layer of neurons and nonlinearity. This is the architecture used to model the digital network, DENN-3, and perform backpropagation during hybrid training.

is due to the introduction of a nonlinearity when intensity detection is performed at the end of the network. Measuring the intensity of a complex field has the effect of individually squaring the real and imaginary components, and summing them. This is equivalent to applying an activation function (with square nonlinearity), and an additional network layer (with fixed binary weights).

We use this idea to build a single-layer complex-valued ONN (ONN-3) that outperforms a linear classifier. A similar complex-valued ONN was recently demonstrated on a photonic integrated circuit [59], although with much smaller

layer sizes. Similar to ONN-1, we use our optical multiplier to connect the 100 input neurons x_i, to the 10 output neurons $z_j^{(1)}$, with complex weights $w_{ij}^{(1)}$, and we measure the intensity of the output vector I_j. The result is given by

$$I_j = \left|\sum_i w_{ji} x_i\right|^2 = \left(\sum_i \mathrm{Re}(w_{ji}) x_i\right)^2 + \left(\sum_i \mathrm{Im}(w_{ji}) x_i\right)^2. \tag{3.9}$$

The softmax function is digitally applied to the intensity vector, giving the network output y_j. This ONN architecture is shown in 3.16(a).

We can reinterpret the network as an equivalent real-valued network as shown in 3.16(b). A real-valued weight matrix $\tilde{w}_{ji}^{(1)}$ connects the 100 inputs to 20 hidden layer neurons. A square nonlinearity is applied to give hidden layer activations $a_j^{(1)}$. A second layer weight matrix $\tilde{w}_{jk}^{(2)}$, which is fixed with values of 0 and 1, connects pairs of hidden layer neurons to create 10 output neurons $z_k^{(2)}$. These output neurons will take exactly the values of I_j in the ONN. Softmax is applied to give the network output y_k. When presented in this architecture the nonlinear form of the network is clear, which shows why the network can outperform a linear classifier.

During hybrid training of ONN-3, both the forward activations and the backward errors are derived from the results of our complex-valued optical MVM. The error vector is computed by digital backpropagation but requires the gradient of the square nonlinearity, which is proportional to $z_j^{(1)}$, the complex-valued output of the MVM. Therefore we do not measure the intensity I_j directly, but use our coherent detection scheme, detailed in Section 3.1.4, to measure and record the real and imaginary components. The activation is then calculated by digitally applying the modulus square.

We plot the learning curves for hybrid training of ONN-3 in Fig. 3.17(a), as well as the curves for an equivalent all-digital network DENN-3, and the linear classifier DENN-1 for reference. Again we see the accuracy quickly improves as the weights converge, with ONN-3 achieving an accuracy of 93.6% on the final validation set, above the linear classifier limit of 92.2%. With just a single 'linear' layer, this demonstrates the advantage of using complex-valued weights. However, the final accuracy is still slightly below the digital equivalent DENN-3.

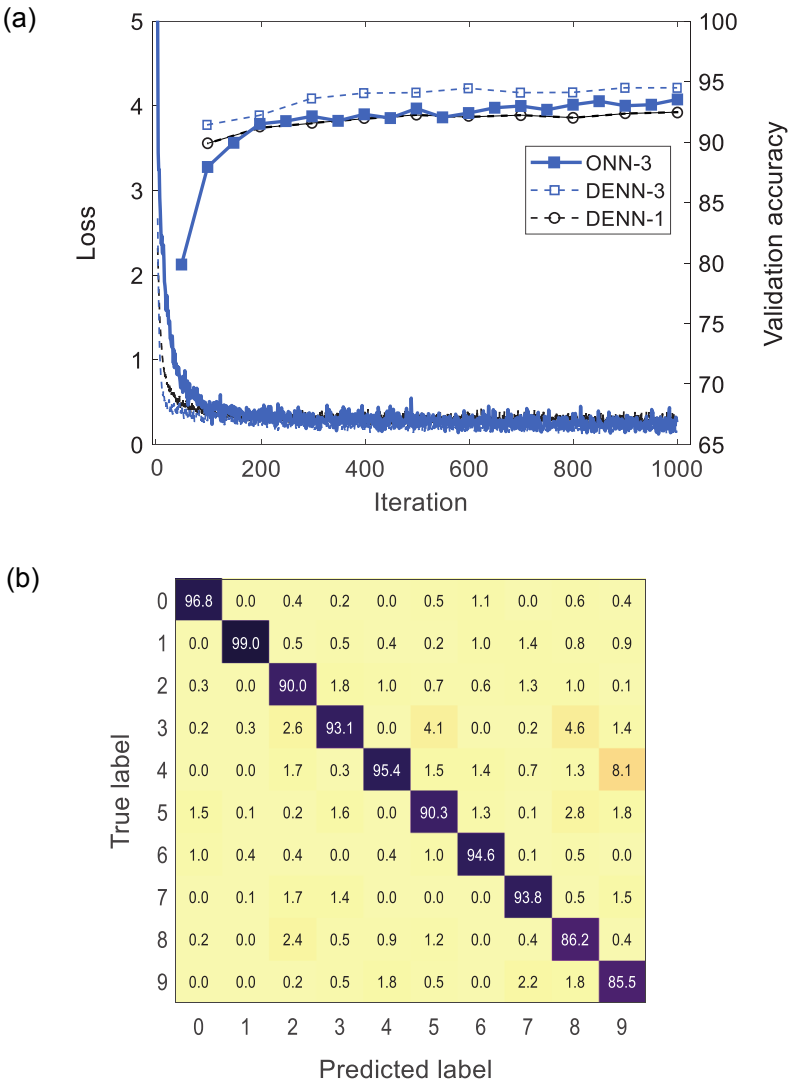


Figure 3.17: Learning curves and confusion matrix for ONN-3. (a) Learning curves for hybrid training of ONN-3, showing loss after each iteration and validation accuracy after every 50 iterations. Also shown are curves for the equivalent complex-valued digital network DENN-3, and the linear classifier DENN-1. The linear classification accuracy limit for this dataset was empirically found to be 92.2%. (b) Confusion matrix for the test set of ONN-3. Figure adapted with permission from [137] © Optica Publishing Group.

Finally, we perform optical inference on the test set, measuring the output intensities I_j directly, without need for the coherent detection scheme. This gave a final validation accuracy of 92.2%, and the confusion matrix is shown in Fig. 3.17(b). Although this does not surpass the linear limit, it does outperform ONN-1 and DENN-1, and therefore shows the true advantage of using complex weights with intensity measurement as a nonlinearity. Whilst training requires the complex procedure of coherent detection, once trained the ONN can be used for inference

using straightforward intensity detection, and achieving higher classification accuracy than real-valued weights alone.

3.2.5 Optical calculation of the error vector

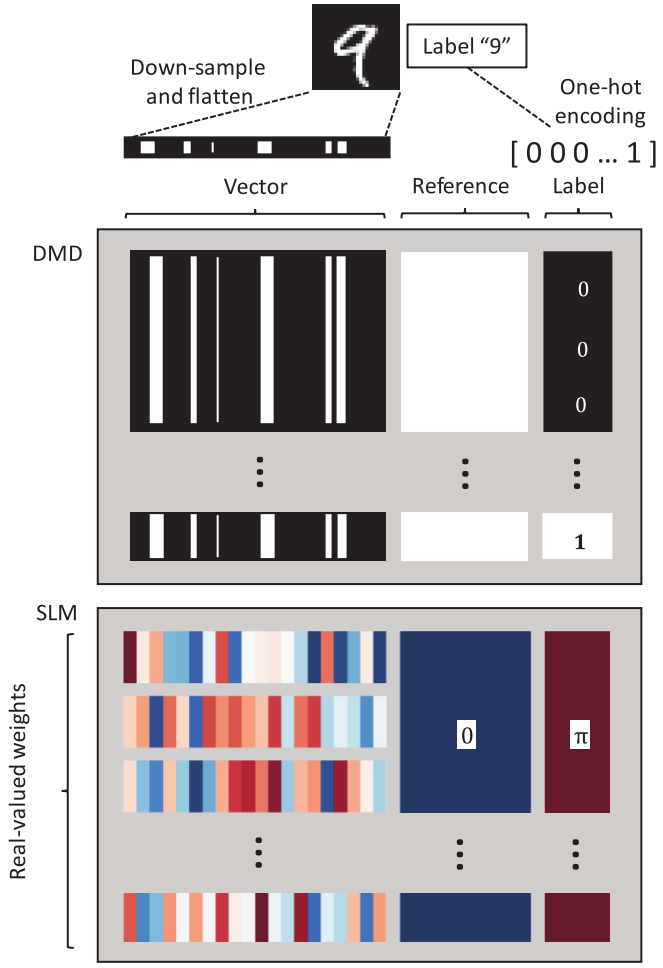


Figure 3.18: Schematic to show DMD and LC-SLM patterns used to perform optical calculation of the error vector. Figure reprinted with permission from [137] © Optica Publishing Group.

Our coherent detection scheme relies on coherent interference between the reference field and our signal, using the cylindrical lens to sum the two beams. We used this as inspiration to demonstrate how we can directly calculate the output error vector, δ^L, without digital computation. We consider a linear classifier like ONN-1, and change the loss function from categorical cross-entropy to mean-

squared error (MSE):

$$\mathcal{L} = \frac{1}{2}\left(z_j^{(1)} - y_j\right)^2. \tag{3.10}$$

Then the output error vector is

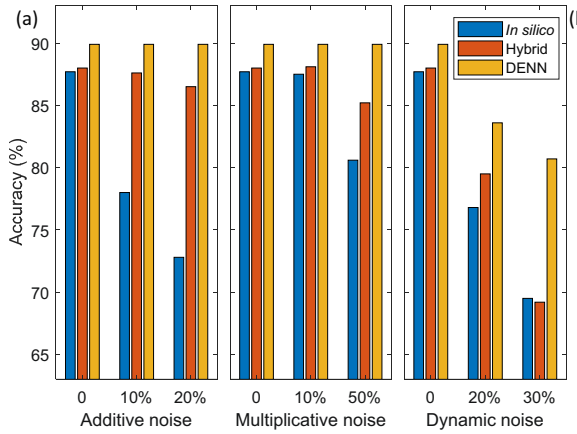
$$\delta_j^{(1)} = \frac{\partial \mathcal{L}}{\partial z_j^{(1)}} = z_j^{(1)} - y_j. \tag{3.11}$$

We can therefore use interference between our MVM output and an optically encoded label to calculate the error vector. To achieve this, we use a third active region on the DMD to encode the label, in addition to the MVM signal and reference region. Since the label is encoded as a one-hot vector, it takes only binary values, and the DMD can encode the input vector and label simultaneously. Then the respective region on the LC-SLM is set to uniform reflection, just as for the reference region, but with a π phase shift relative to the signal and reference, in order to encode the necessary negative sign. A schematic of the three regions on the DMD and LC-SLM that encode the three fields is shown in Fig. 3.18.

Intensity measurement at the output can be used to directly calculate the real-valued error (3.11). To calculate the weight updates of a linear classifier, all that remains is to take the outer product with the inputs x. In this way, almost the entire training procedure is performed optically. We retrained ONN-1 using this scheme, and achieved a validation accuracy of 83.3%, and inference test accuracy of 83.4%. The classification accuracy was lower than previously measured, but this is mainly attributed to the change of loss function, since MSE is not well suited to classification problems. To confirm this, we digitally retrained DENN-1 using MSE loss, which gave a test accuracy of 85.7%, only marginally better than our ONN.

This experiment was a good demonstration of how optics can be used in different aspects of the neural network training. However it is not straightforward to generalise this method of optical label encoding and destructive interference to deeper, nonlinear networks, nor to other loss and activation functions where the error term is not as simple as with MSE. As such we did not pursue optical calculation of the output error beyond this simple proof-of-principle demonstration.

3.2.6 Comparing hybrid and *in silico* training



(b)

	Hybrid training	*In silico* training	DENN
No additional noise	88.0%	87.7%	89.9%
10% static additive noise	87.6%	78.0%	89.9%
20% static additive noise	86.5%	72.8%	89.9%
10% static multiplicative noise	88.1%	87.5%	89.9%
50% static multiplicative noise	85.2%	80.6%	89.9%
20% dynamic additive noise	79.5%	76.8%	83.6%
30% dynamic additive noise	69.2%	69.5%	80.7%

Figure 3.19: Results showing the impact of different types of imperfection during network training of ONN-1. (a) Comparison of hybrid training (red bars) and offline training (blue bars) of ONN-1 in presence of experimental errors. Results for digital training of DENN-1 with equivalent errors are also shown (yellow bars). Left, middle, and right panels show effects of static additive noise, static multiplicative noise, and dynamic additive noise, respectively. (b) List of classifier accuracy achieved for different noise levels. Figure reprinted with permission from [137] © Optica Publishing Group.

As described in the introduction, one critical aspect of in-the-loop training schemes, such as our hybrid training of ONNs, is the ability to overcome the 'reality-gap' between the physical hardware and a digital model. We performed offline (or *in silico*) training with ONN-1, by training an equivalent DENN on a PC (including identical input data, numbers of neurons, hyperparameters and constraints) and uploaded the weights to our ONN. Surprisingly, we found that using the optical system to perform inference on the test set reached 87.7% classification accuracy, essentially as good as when using the weights found with hybrid training at 88.0%. To emphasize, during this offline training we don't model the physical system, and we don't introduce any artificial noises to force the network to be robust against experimental imperfections, we simply use a DENN of the same architecture.

This indicates we have a well-calibrated system, having eliminated most aberrations and systematic errors, and demonstrates the low RMSE of our MVM is suitable for use in an ONN. However, it is likely that larger and deeper future ONNs will not achieve such good precision, and analog hardware is liable to vary

in precision over time and between devices. Therefore to explore how robust our hybrid training scheme is to such errors, especially in comparison to offline training, we retrained ONN-1 whilst purposefully introducing various types of imperfection into the experiment.

Using the LC-SLM we introduce to the setup three different types of error attributed to the weight matrix: static additive noise, static multiplicative noise and dynamic additive noise. Static additive noise might arise from ambient light, imperfections such as dust on the optics, or imprecise device calibration. Static multiplicative noise may be caused by non-uniform transmission of different optical channels, imperfect interference, different responses of camera pixels, and so on. Dynamic noise, by which we mean errors that fluctuate over time, may arise from non-linear device calibration, environmental fluctuations such as vibrations and laser power drift, and crosstalk from changing neighbouring matrix elements.

The first imperfection, static additive noise, is defined as a random bias $W_{ji} \rightarrow W_{ji} + \epsilon_{ji}, \epsilon \in N(0, \sigma)$ applied to each weight matrix element, which remains unchanged during the training and testing. The noise level is defined by the standard deviation σ, which is normalized by defining the maximum value of all weight matrix elements as one, i.e. $W_{ji} \in [-1, 1]$. In this test we use values of $\sigma = 0.1$ and $\sigma = 0.2$, or 10% and 20% noise levels. We randomly sample the bias which is then fixed during the entire training and testing process.

As shown in the left panel of Fig. 3.19(a), hybrid training is fairly robust to such static additive noise, because the imperfection is included in every iteration of training, and the network has 'awareness' of the systematic error. Meanwhile, the accuracy of offline training drops to 72.8% at the 20% noise level, significantly degraded. This is a clear demonstration of the requirement for in-the-loop training schemes when dealing with imperfect physical hardware.

The second common imperfection, static multiplicative noise, is defined as $W_{ji} \rightarrow W_{ji} \cdot \eta_{ji}, \eta \in N(1, \sigma)$, where the noise level is again indicated by the standard deviation σ and the multiplicative factor remains fixed throughout training and

testing. Fig. 3.19(a) shows a similar, although less severe, degradation in offline training performance compared to hybrid, reducing to 80.6% at 50% noise level.

The last major type of imperfection we explore is dynamic noise. In the experiment we model such dynamic noise by additive noise (as defined above) applied to each weight element, that is randomly re-sampled before each weight update of the training. Our results show that the ONN is sensitive to such random dynamic noise, using weights from either hybrid or offline training. In both cases the accuracy drops to about 69% at 30% noise level.

We simulated all of these experiments with an equivalent DENN, and the results are also shown in Fig. 3.19(a). For both types of static noise, the DENN is able to maintain the maximum classifier accuracy, but for random dynamic noise, the performance also degrades. It is therefore clear that hybrid training, and in-the-loop training schemes more generally, are unable to correct for these types of error, because the fluctuating noise limits the precision of each gradient descent step and the error can not be systematically compensated for by additional weight updates.

3.3 Conclusion

In this chapter we have explored the application of our optical multiplier as the linear layer in various types of optical neural network. We took advantage of the ability to encode information in both the amplitude and phase of a coherent beam to demonstrate real- and complex-valued ONNs, and showed that utilising complex values can improve classification accuracy if some form of nonlinearity, such as intensity detection, is introduced.

We implemented hybrid training, a form of hardware-in-the-loop training, using the optical system in every training iteration. This was an important first step towards the larger goal of end-to-end optical training of an ONN. By artificially introducing different types of error to our optical multiplier, we were able to show that hybrid training can outperform offline training for certain types of systematic error resulting from imperfections and aberrations in the physical system.

We have validated our optical MVM has sufficient precision for use in an ONN by achieving good classification accuracy on the MNIST classification task. However it is important to acknowledge that this is a relatively simple benchmark. For future implementations of ONNs, and analog hardware in general, it will be important to explore if this level of precision is sufficient when tackling more realistic workloads, assuming a similar level of precision can be maintained as the ONN size is scaled up.

Furthermore, we have so far only utilised a single optical multiplier to perform one linear layer of the ONN. In order to create a deep ONN, it is necessary to cascade multiple optical MVM layers without optical-electronic conversion. Future deep ONNs with many layers must consider how errors in the multiplication cascade between layers. Additionally, in order to extend our hybrid training scheme towards end-to-end optical training, it is necessary to perform both forward and backward passes of each training iteration with the optical system. This is the focus of the next chapter.

In this chapter we have seen that hybrid training can overcome the 'reality gap' between physical hardware and digital simulation, but we have also shown that classification accuracy still degrades as error levels are increased in the optical system. As we extend the number of calculations in the network training that are performed optically, maintaining the multiplier calibration and MVM precision across all layers will be crucial to successfully demonstrating optical training.

4

Two-layer ONN and Optical Backpropagation

Contents

To demonstrate a fully-optical implementation of the backpropagation algorithm, we must first construct a multi-layer ONN. In this chapter I explain how we experimentally achieved a two-layer network without the need for opto-electronic conversion. With this network, we then perform our optical backpropagation scheme, for the first time showing how every step of the algorithm can be performed with optics. For now we omit the nonlinear activation at the hidden layer, which is introduced in Ch. 5. We apply our network to a simple classification problem to evaluate the performance against offline training, and find that optically training our ONN indeed achieves higher inference accuracy.

4.1 Two-layer ONN

4.1.1 Cascaded MVM concept

Recall our optical MVM scheme consists of the following:

1. Encoding N-dimensional vector, and fan-out,
2. element-wise multiplication by $M \times N$ matrix,
3. fan-in, and selection of zero-order frequency component to yield M-dimensional vector.

We now refer to this as MVM1, and its output vector, encoded in the field of the optical beam, can naturally become the input vector to another MVM, referred to as MVM2, without the need for opto-electronic conversion. The 'geometry' of MVM2 will look identical to the first, rotated by 90°, as depicted conceptually in Fig. 4.1.

We extend the necessary steps for our cascaded MVM system with:

4. second fan-out,
5. element-wise multiplication by $L \times M$ matrix,
6. second fan-in, and selection of zero-order frequency component to yield L-dimensional vector.

We use such a system as the basis of a two-layer ONN, and so adopt the relevant notation and terminology as follows. The first vector and matrix are the input-vector $a_i^{(0)}$, and first-layer weight matrix $W_{ji}^{(1)}$. The result of MVM1 is the hidden-layer vector

$$z_j^{(1)} = \sum_i W_{ji}^{(1)} a_i^{(0)}, \tag{4.1}$$

which also forms the input to MVM2 $a_j^{(1)} \equiv z_j^{(1)}$ (we neglect the nonlinear activation function for now). This is multiplied by the second-layer weight matrix $W_{kj}^{(2)}$, and finally the result of MVM2 is the output vector

$$z_k^{(2)} = \sum_j W_{kj}^{(2)} a_j^{(1)}. \tag{4.2}$$

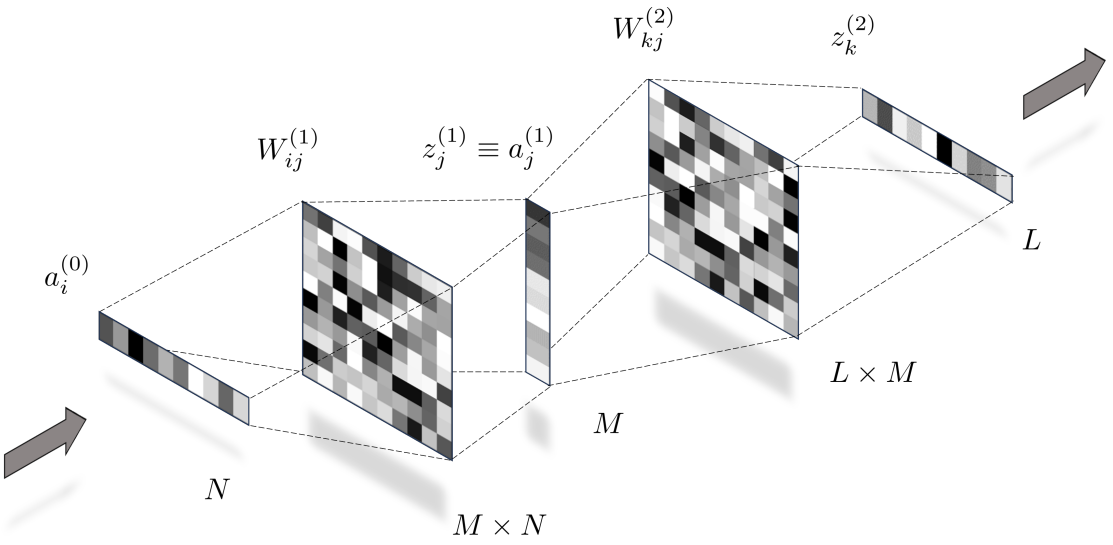


Figure 4.1: Conceptual diagram of two cascaded MVMs. The geometry of the vector fan-out and fan-in is shown, and vector and matrix dimensions and labels used throughout the main text are shown.

We used our existing experiment to perform MVM1, and extended the setup to perform MVM2 and construct the two-layer ONN. Together the cascaded MVMs constitute a single optical system, shown conceptually in Fig 4.2(a). We refer to the individual devices used in each layer as DMD1 and SLM1, and SLM2.

In Fig. 4.2(b) we illustrate how the beam looks throughout the two-layer ONN, with images taken from experiment. The numbers indicate the (conceptual) position in the experiment in (a). In this example we use $N = 5$, $M = 10$ and $L = 6$, and perform parallel vector-vector multiplication (p-VVM) in the first layer. MVM1 uses the original combination of DMD1, SLM1 and cylindrical lenses, which we label CL-1. Images (1) and (2) show the field immediately after DMD1 and SLM1 respectively, and image (3) shows the field after fan-in by CL-1.

An important modification to the original experiment had to be made. The MVM result is encoded in only the zero-order frequency component of the beam, which forms a narrow straight line at the centre of the output plane. So far, when implementing just a single optical MVM, we used a camera to detect the intensity of the entire output plane, and then digitally processed the image to measure the MVM result. However, in this cascaded MVM system this is not possible, as the hidden-layer vector must remain encoded in the optical beam. We therefore replaced

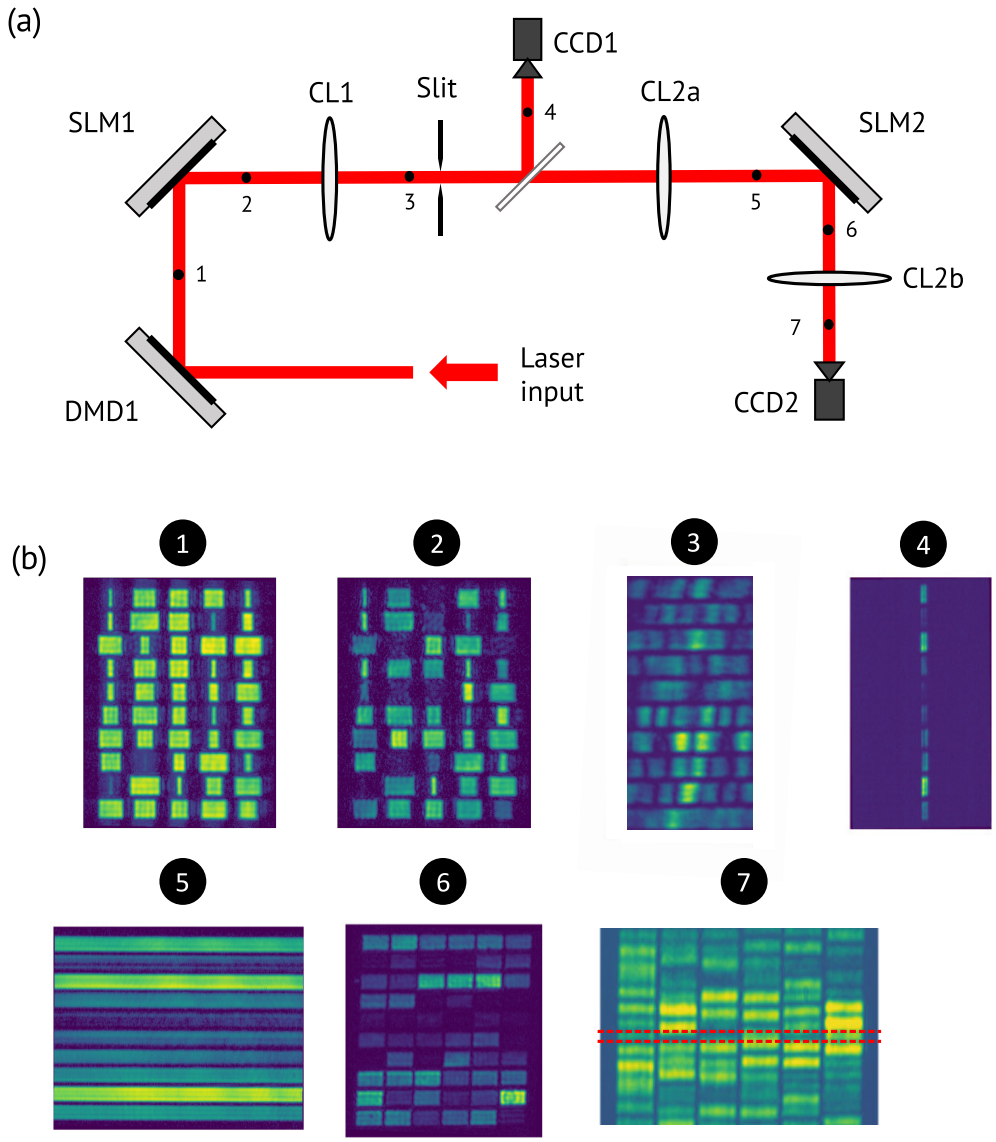


Figure 4.2: Simplified experiment scheme for two-layer cascaded MVM and experiment images. (a) Diagram showing the critical components of the cascaded MVM. (b) Intensity images taken from experiment for one example cascaded MVM with $N = 5$, $M = 10$ and $L = 6$. The numbers correspond to the position in the beam path the images were taken, indicated in the diagram.

the camera with a narrow adjustable slit, precisely positioned so as to only admit the required zero-order component of the field. The slit was mounted such that it could be precisely translated and rotated, and the width could be adjusted. Further experiment details are given in the next section.

After the slit, we use a beam splitter to tap-off a portion of the beam, which is imaged onto a camera, CCD1, to measure the hidden-layer vector $a^{(1)}$. Image

(4) in Fig. 4.2(b) shows the field immediately after the narrow slit, demonstrating how the transmitted field approximates very well the admission of the zero-order frequency component only.

Continuing in the main signal path, a new cylindrical lens set CL-2a is used to perform optical fan-out. The orientation of lenses in this set is identical to CL-1, such that the combination of narrow slit and cylindrical lens acts to produce a very broad beam, uniform in the horizontal direction but still encoding the vector $a^{(1)}$ in the vertical. The example of this broadened beam is shown in image (5). This can be interpreted as creating many copies of the hidden-layer vector, which are then imaged onto a new LC-SLM, SLM2. This encodes the real-valued second-layer weight matrix $W^{(2)}$, and performs element-wise multiplication, resulting in the field shown in image (6). A third and final set of cylindrical lenses, CL-2b, performs optical fan-in, this time in the vertical direction, i.e. in the direction orthogonal to CL-1. This completes MVM2 and a new camera, CCD2, is used to measure the output field, depicted in image (7). In order to simplify the experiment, the final output vector $z^{(2)}$ was measured without a slit, with the zero-order component extracted by digital processing as before.

Together, the two lens sets CL-1 and CL-2a act like a 4-f imaging system, with the slit at the Fourier plane. If the slit is removed, the field at the output plane after CL-2a should be an exact image of the input before CL-1. This is shown in Fig. 4.3(a), with images taken from experiment showing the input of CL-1 and the reconstructed image after CL-2a with the slit removed from the experiment. The images are almost identical, although the reconstructed image is clearly titled by a few degrees, due to slight misalignment of the cylindrical lenses. The slit then acts as a filter at the Fourier plane, 'blurring' the image in the horizontal direction. The effect of adding the slit is shown in Fig. 4.3(b).

4.1.2 Narrow slit in experiment

The narrow slit used (Thorlabs VA100) was able to be closed completely, within a specified tolerance of 20 μm. Using the narrow slit created a major experimental

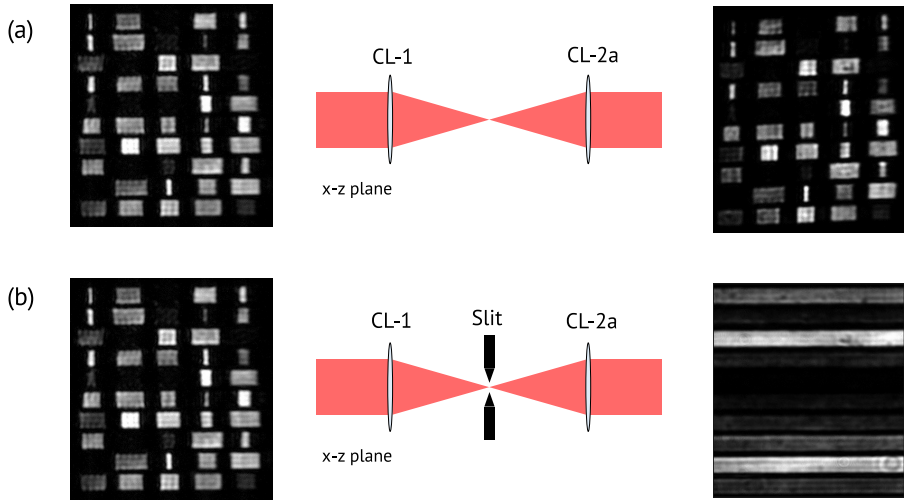


Figure 4.3: Images to show the effect of the narrow slit. (a) Without the narrow slit present, the two sets of cylindrical lenses CL-1 and CL-2a act as a 4-f imaging system. We show only the central lens of each set in the conceptual diagram. Images are taken from experiment, showing the field pattern for one example random MVM at the image plane before CL-1 and after CL-2a. (b) The narrow slit acts as a spatial filter, selecting only the zero-order spatial frequency (in the x-direction), and CL-2a subsequently fans-out the narrow vector.

difficulty, due to mechanical instability in the setup. The interference pattern at the MVM output plane is a complicated structure of bright and dark fringes, with the MVM result encoded in only the very centre fringe. In many cases the contrast in field amplitude of the central fringe and immediately adjacent fringes was very large, several examples of which can be seen in the example images shown in Fig. 4.2(b). Therefore, with a narrow slit selecting the central fringe, any fluctuation in the beam position relative to the fixed slit can cause an extreme fluctuation in the field transmission, and so a great deal of noise in the measured MVM result.

The greatest source of fluctuation in beam position was due to the mechanical movement of the micro-mirrors in the DMD. Any small angular movement of the micro-mirrors caused a substantial shift in position of the beam at the slit, which was by design at the Fourier plane of the DMD. Since the DMD mirrors are continuously actuated, at 10 kHz, and are sensitive to thermal drifts [159], the beam could be observed to fluctuate significantly at the slit plane. This was measured quantitatively to estimate the range of motion: the entire DMD was illuminated, and the beam passed through a spherical lens with focal length $f = 500$ mm and

imaged by a camera. Many frames were captured, and the centre of the beam spot was numerically found in each frame. The distance moved could be estimated by using the known camera pixel pitch of 3.45 μm. In the worst case, the beam was measured to shift by $\Delta u = \pm 20$ μm over the duration of only 30 s. This translates to an angular displacement of the DMD of approximately ± 40 μrad.

In MVM1, the width of the narrowest central fringe at the slit plane is determined by the Fourier transform of the rectangular aperture of DMD1, described by the sinc function $\sin(\pi x)/\pi x$. More precisely, a rectangular aperture of width w_0 is given by

$$a(x) = \text{rect}\left(\frac{x}{w_0}\right). \tag{4.3}$$

The field distribution at the back focal plane of a lens with focal length f, for a beam with wavelength λ, is then

$$b(u) = \mathscr{F}\left[a(x)\right]\left(\frac{u}{\lambda f}\right) \tag{4.4}$$

$$\propto \text{sinc}\left(\frac{w_0 u}{\lambda f}\right). \tag{4.5}$$

The distance between the zeros of this sinc function is therefore

$$w_1 = \frac{2\lambda f}{w_0}. \tag{4.6}$$

The laser wavelength was $\lambda = 780$nm, and the combination of spherical lenses (for the spatial filter after SLM1) and cylindrical lenses CL-1 combine to give an effective focal length value of $f = 600$mm. The width of SLM1 was $w_0 = 14.5$mm, so we find $w_1 = 54.1\mu$m. In practice the slit must be much narrower than this, to select as close to only the zero-frequency component as possible. The exact width of the slit was set empirically, in order to balance two competing requirements: increasing the precision of the MVM result (requiring a narrower slit) and minimising optical power loss (requiring a wider slit). We found that setting the slit width to approximately a quarter of w_1, so $w_{\text{slit}} = 0.25 w_1$, gave good precision of the MVM1 result, whilst maintaining acceptable optical power for MVM2. However, it is clear that the beam position fluctuation of $\Delta u = \pm 20\mu$m is unfeasible with a value of $w_{\text{slit}} \approx 14\mu$m, which in any case is less than the tolerance of the adjustable slit.

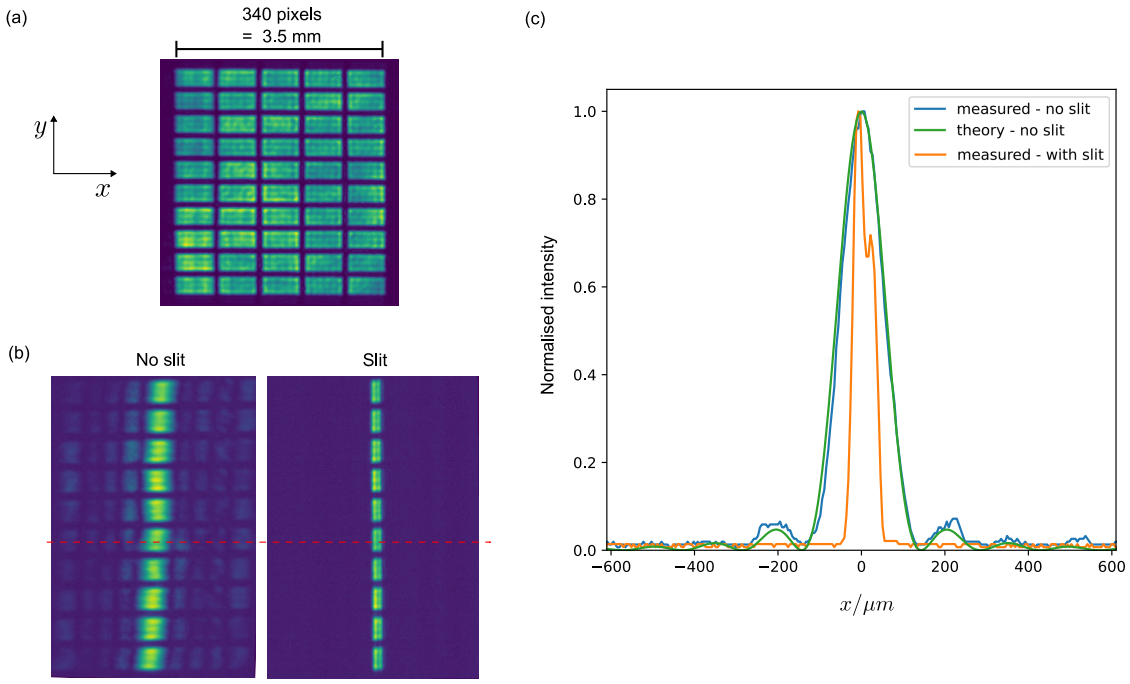


Figure 4.4: Measuring the width of the narrow slit in experiment. (a) Image from experiment of the reduced LC-SLM signal region, with a uniform matrix pattern. (b) Images from experiment after CL-1 has performed a fourier-transform in the horizontal direction, with and without the slit present. (c) Cross-section of the intensity measured along the red dashed line. A numerically-fit theory curve of the sinc pattern is also shown.

There were two possibilities to overcome this problem. The first was to reduce the beam fluctuation as much as possible. This option was explored, with a scheme to actively stabilise the beam position using a piezo-actuated mirror and a quadrant detector. Ultimately this was not viable, and instead we opted to increase w_1 such that the beam fluctuations were inconsequential. The easiest method to achieve this was by reducing the active area of DMD1 and SLM1, thereby reducing w_0, and proportionally increasing w_1. A new active region of DMD1 and SLM1, referred to as the signal region, was 340 pixels wide, or approximately 3.5mm. In this case, we calculate a theoretical value of $w_1^{(\text{theory})} = 266\mu\text{m}$, and $w_{\text{slit}} \approx 60\mu\text{m}$, such that the beam fluctuation is far less impactful on the transmitted field amplitude. In Fig 4.4(a) we show an image from experiment of a uniform matrix with $N = 5$ and $M = 10$ where all elements are set to maximum, encoded with the new reduced signal region. In (b) we show the images recorded at the hidden-layer output, without any slit in place (left) and with the slit closed (right). The sinc pattern is

clearly visible in the horizontal direction, and in (c) we plot the cross-section of the measured image, with and without the slit, along the dashed-red line shown. We also plot a theoretical sinc function numerically fit to the experimentally measured data. By counting camera pixels and using the known camera pixel pitch of 4.5μm, we determine $w_1^{(\text{meas})} = 274 \pm 9\mu$m, which agrees with the theoretical value.

4.2 Optical backpropagation

4.2.1 Concept

Whilst expanding the experiment to construct the two-layer ONN, we also constructed the optical system necessary to implement our optical backpropagation scheme. Recall that network training requires calculating the gradient of the loss function $\mathcal{L}$ with respect to all the network weights $W^{(l)}$, given by

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^{(l)}} = \delta_j^{(l)} a_i^{(l-1)}, \tag{4.7}$$

requiring us to calculate the activations $a^{(l)}$ and errors $\delta^{(l)}$ at every layer of the network. In our hybrid training scheme, the activations were calculated optically from the forward propagation of our signal through the ONN, whilst the errors were found on a digital model. The ambition of our optical training scheme is very simple: to calculate both the activations and the errors optically, by performing the backpropagation algorithm in optics.

To achieve this, we use a second optical beam encoding the error, that passes through the setup in the reverse direction. In this sense, the mathematical algorithm takes on its literal physical meaning: the optical error beam *propagates backwards*. The beam can be introduced to the system using a simple beam splitter, and aligned so as to exactly match the path of the forward beam, in the reverse direction. Here we outline how our optical backpropagation scheme works in the case of an ONN with two linear layers, for now neglecting the nonlinear activation function at the hidden layer.

Output-layer error

Recall the algorithm begins by calculating the error at the final layer directly from the loss function:

$$\delta_k^{(2)} = \frac{\partial \mathcal{L}}{\partial z_k^{(2)}} = \frac{\partial \mathcal{L}}{\partial y_n} \frac{\partial y_n}{\partial z_k^{(2)}}. \tag{4.8}$$

In our case, $z^{(2)}$ is a vector of dimension L, the final-layer MVM result of the ONN for an input example x, and y is the final ONN output prediction after application of an activation function on $z^{(2)}$. Further recall that for classification problems, this is commonly the softmax function, which has the effect of normalising all vector components to sum to one, allowing them to be interpreted as probabilities. When used in conjunction with categorical cross-entropy (CCE) loss function, and for our two-layer ONN, (4.8) simplifies nicely to

$$\delta_k^{(2)} = y_k - t_k, \tag{4.9}$$

where t is the label associated with input example x (see Appendix A).

In our scheme, $z^{(2)}$ is found after optical forward propagation through the ONN, and measured by the camera CCD2. Then a digital computer is used to calculate y and $\delta^{(2)}$, which is re-encoded in the backward propagating beam. This optical-electronic-optical interconversion is not optimal, however it greatly simplifies the experiment, as it decouples the forward and backward propagating beams. Whilst it is possible to directly generate the error $\delta^{(2)}$ optically using destructive interference, as we showed previously, it would be extremely difficult to retroreflect this beam back through the network. In particular, various sources of optical loss throughout the forward pass of the ONN resulted in a very weak beam at the output, and the issues of mechanical instability and beam fluctuations at the slit plane would be accentuated as the path length is effectively doubled.

Hidden-layer error

The error vector $\delta^{(2)}$, a vector of dimension L, is re-encoded in the backward optical beam using a new DMD, DMD2. The next stage of the backpropagation

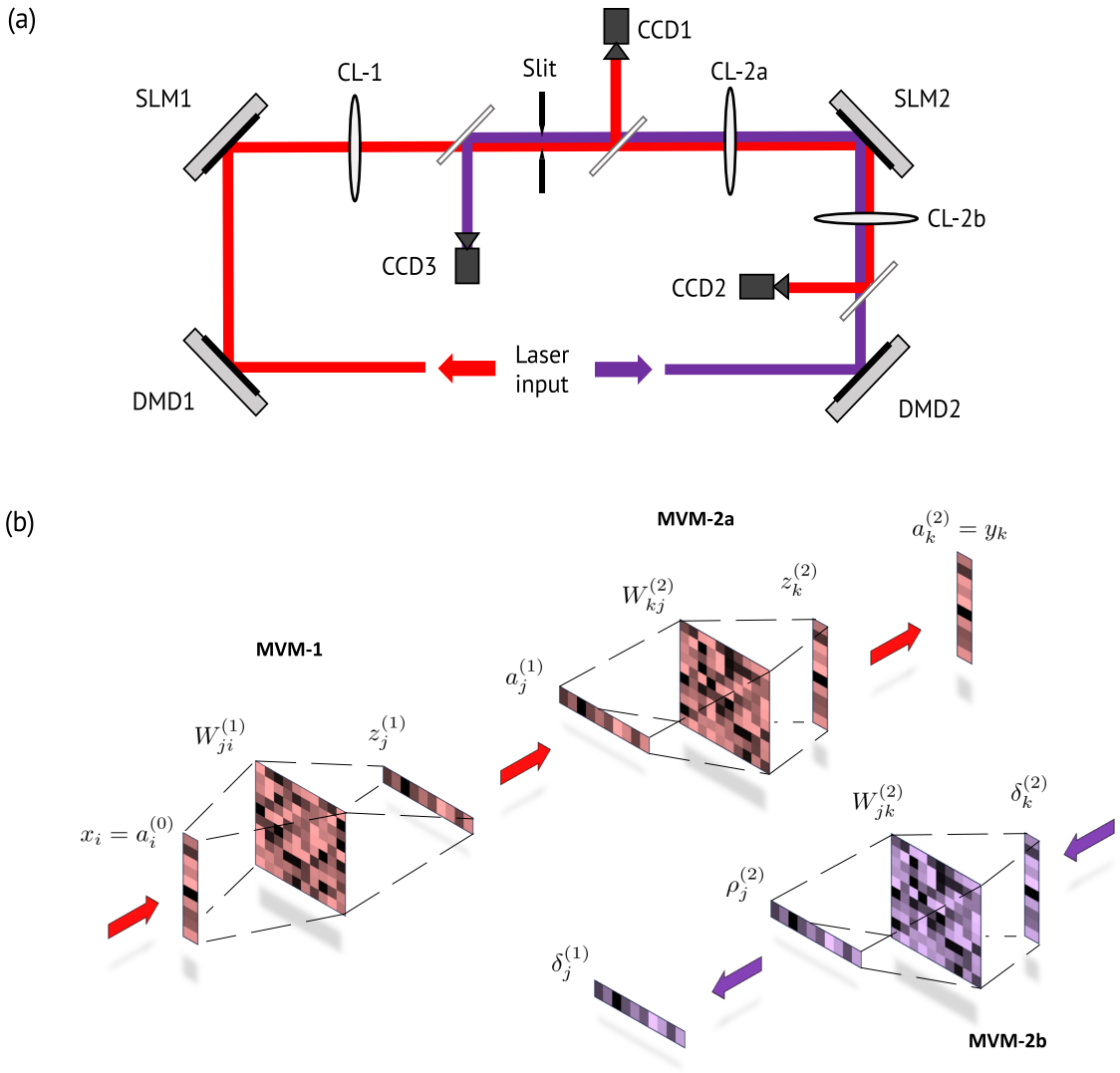


Figure 4.5: Conceptual diagrams for backpropagation in our two-layer ONN. (a) Simplified experiment scheme, showing the critical components added to the setup to implement optical backpropagation. (b) Visualising the beam geometry for the MVMs in forward and backward directions, with the associated neural network vector and matrix labels.

algorithm is to calculate the error vector at the first layer, $\delta^{(1)}$. This is achieved by performing an MVM of $\delta^{(2)}$ with the second weight matrix, $W^{(2)}$, and multiplying by the derivative of the hidden-layer activation:

$$\delta_j^{(1)} = \rho_j^{(2)} \cdot g'\left(z_j^{(1)}\right), \tag{4.10}$$

where

$$\rho_j^{(2)} = \sum_k W_{jk}^{(2)} \delta_k^{(2)}. \tag{4.11}$$

For now we assume a linear network, without the hidden-layer activation function $g(\cdot)$. We introduce this key aspect of our optical training scheme in Chapter 5, however for now we can neglect the derivative term, and get the simple relation $\delta^{(1)} = W^{(2)^T} \cdot \delta^{(2)}$, where T denotes the transpose of the matrix.

We can achieve this new MVM optically by performing the same physical operations as we did in the forward direction. Recall we performed MVM2,

$$z_k^{(2)} = \sum_j W_{kj}^{(2)} a_j^{(1)}, \tag{4.12}$$

by using cylindrical lenses CL-2a to fan-out the vector $a^{(1)}$, SLM2 to element-wise multiply by $W^{(2)}$, and using CL-2b to perform fan-in. We now do similarly in the backwards direction, whereby lenses CL-2b perform fan-out of vector $\delta^{(2)}$, SLM2 performs element-wise multiplication by $W^{(2)^T}$, and CL-2a performs fan-in to yield $\delta^{(1)}$.

We show a conceptual scheme of how this bi-directional MVM works with our two-layer ONN in Fig. 4.5(a) and (b). After generating the backward beam, and encoding the error with DMD2, the backward beam passes through a beam splitter and is aligned perfectly with the forward beam. The beam splitter allows the forward beam to be detected by CCD2. Similarly, a third beam splitter is added to the path just before the slit, allowing the backward beam to be tapped-off and detected by a third camera CCD3, which is used to measure the result $\delta^{(1)}$.

In Fig. 4.5(b) we conceptually show how the beam geometry works to perform MVM in the forward and backward directions. In particular we see how the lenses CL-2a and CL-2b each perform opposite roles of fan-in and fan-out in either direction. The key idea is that the geometry of the setup means we do not have to change or update any part of the system to accommodate the backward MVM calculation. In particular, displaying the appropriate pattern on SLM2 encodes the correct matrix in both forward and backward MVMs, with the transpose operation automatically fulfilled. We refer to the forward and backward MVMs in the second layer as MVM2a and MVM2b respectively.

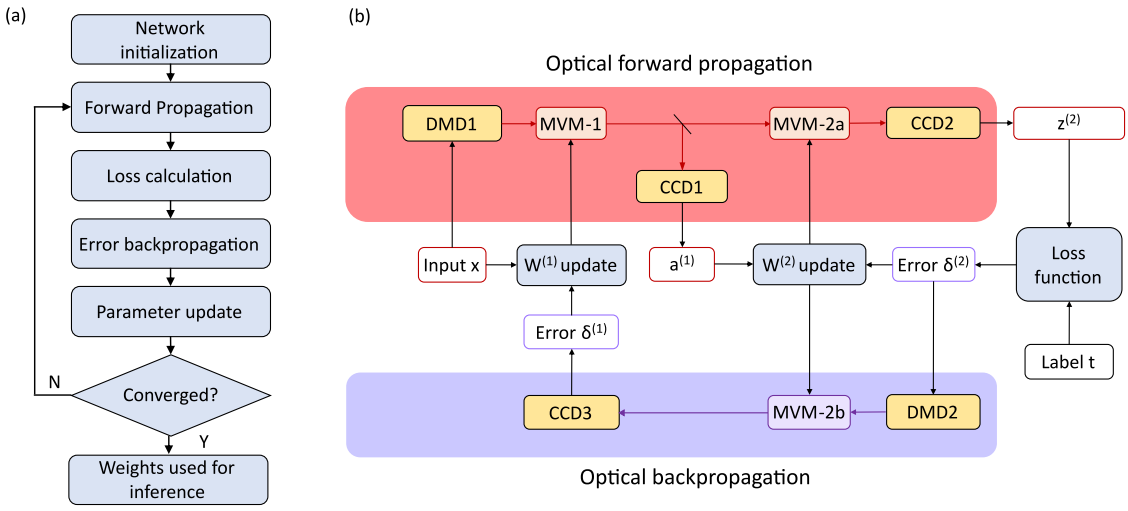


Figure 4.6: Flow charts to show the optical training procedure. (a) General supervised learning procedure to train a neural network. (b) Optical training scheme, showing the optical information flow in both forwards and backwards directions.

Updating the weights

We optically calculate three vectors, $a^{(1)}$, $z^{(2)}$ and $\delta^{(1)}$, which are detected by cameras CCD1, CCD2, and CCD3 respectively. Additionally, we have used $z^{(2)}$ to digitally calculate the output error $\delta^{(2)}$, and have used the known inputs $x \equiv a^{(0)}$. We can now use these vectors to find the gradients of the loss function with respect to both weight matrices, using

$$\frac{\partial \mathcal{L}}{\partial W_{ji}^{(1)}} = \delta_j^{(1)} a_i^{(0)}, \tag{4.13}$$

and

$$\frac{\partial \mathcal{L}}{\partial W_{kj}^{(2)}} = \delta_k^{(2)} a_j^{(1)}. \tag{4.14}$$

These gradients are used to update the weights according to

$$W^{(l)} \leftarrow W^{(l)} - \eta \left(\frac{\partial \mathcal{L}}{\partial W^{(l)}} \right), \tag{4.15}$$

for some chosen learning rate η. The physical weight update is performed by changing the electrically-addressed LC-SLMs. Once the LC-SLMs are correctly displaying the updated weights, a new training batch can be run by repeating the entire procedure with new inputs. The entire training sequence is summarised in Fig. 4.6(a), and the

information flow for a single training batch is depicted in Fig. 4.6(b), highlighting how the majority of the calculation is performed in the optical domain, with digital electronics only required for the loss function and physical parameter update.

4.2.2 Two-layer ONN with backpropagation - methods

Both forward and backward beams were generated from the same laser, and a combination of polarising beam splitters (PBS) and half-waveplates were used to control the relative intensity of the two beams. A detailed schematic of the experiment is shown in Fig. 4.7, including all spherical and cylindrical lenses used to prevent unwanted diffraction of the beam. Not all mirrors used in experiment are shown, and two simplifications are made: the DMDs and SLMs are depicted as transmissive, when in reality they are reflective as before, and additional optical elements used to create alternative paths for camera measurements and monitoring are not shown. We also summarise all the key components in Tab. 4.1.

Encoding vectors

Whilst we individually refer to DMD1 and DMD2 encoding the input and error vectors, in practice two halves of a single DMD were used. We still refer to the individual regions of the DMD encoding each vector as DMD1 and DMD2. Furthermore, the entire DMD plane is imaged onto the entire SLM1 plane. The forward beam perfectly maps DMD1 to the active signal region of SLM1, encoding $W^{(1)}$, to perform the first part of MVM1. Meanwhile, after reflecting from DMD2 to encode the amplitude of $\delta^{(2)}$, the backward beam also reflects from SLM1. This region of the SLM is used to modulate only the phase of the backward beam, in order to encode the sign of $\delta^{(2)}$, as explained below. We depict this first part of the experiment in Fig. 4.8. Using a single device to encode both forward and backward beams simplified the data acquisition during optical training, removing the need for external synchronisation of both signals.

The encoding method for the positive-only vector x and real-valued weight matrix $W^{(1)}$ is exactly as before. The vector is encoded by varying the number of physical

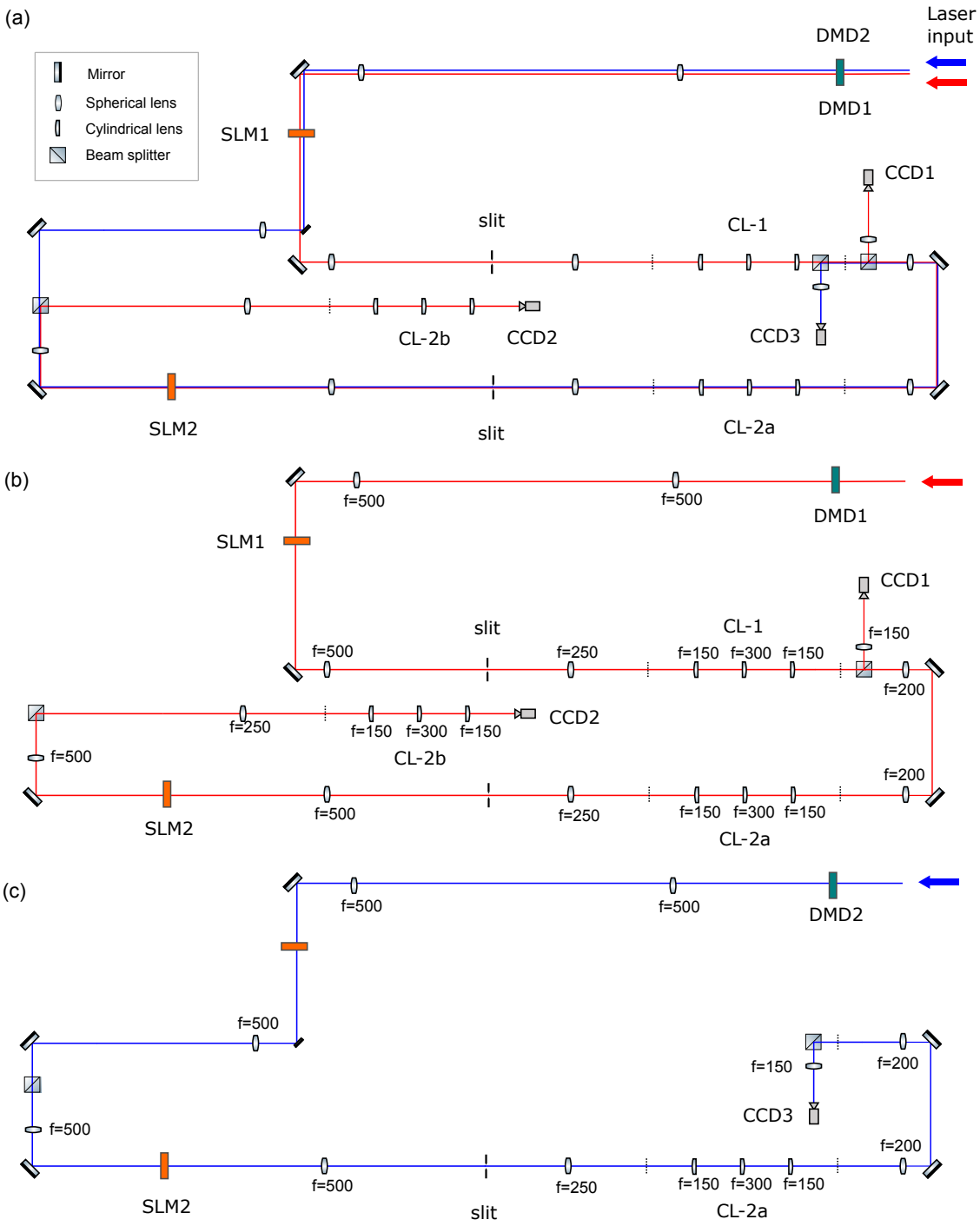


Figure 4.7: Full experiment diagram for two-layer ONN with optical backpropagation. These diagrams show all lenses used in experiment, but simplify the beam path by omitting mirrors, and in particular, the DMD and SLMs are depicted as transmissive, not reflective. (a) Both forward and backward beams. (b) Forward beam only. (c) Backward beam only.

pixels illuminated within each 'logical' pixel on DMD1, and the weights are encoded by adopting the phase-grating technique in the signal region of SLM1. However,

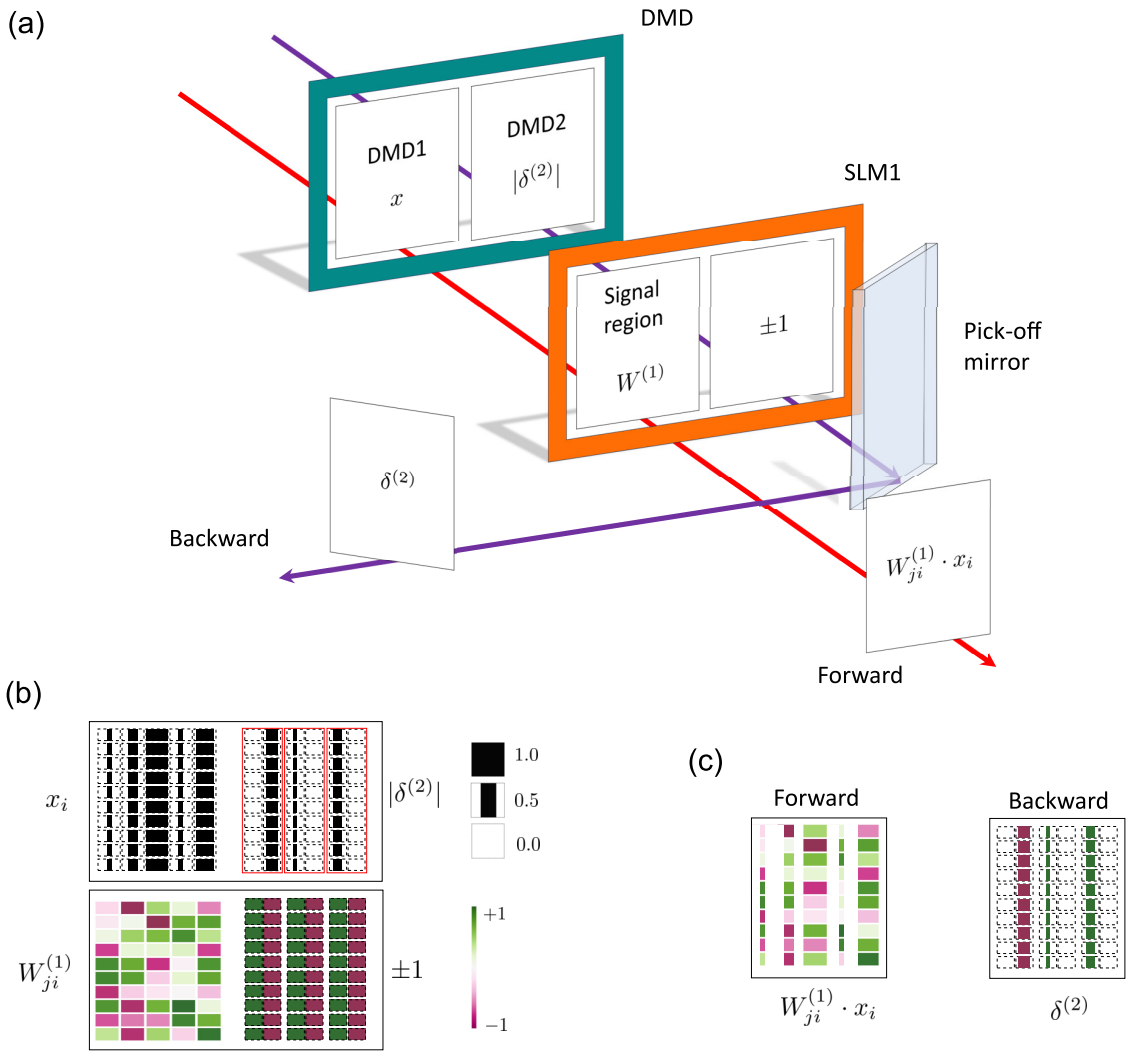


Figure 4.8: Method of encoding real-valued error vector with DMD and LC-SLM. (a) Each half of the DMD and SLM1 are used to encode information in the forward and backward beams. The two beams are then separated by a pick-off mirror. (b) Example patterns displayed on the DMD and SLM. (c) The fields generated in each direction after the pick-off mirror.

because the signal region was reduced to 340 pixels, whilst maintaining the grating period at 10 pixels, the maximum input dimension achievable was $N = 30$. This allowed us to accommodate square images of size 5×5, flattened to a vector of $N = 25$. We used $M = 10$ hidden layer neurons, and a maximum of $L = 3$ output neurons.

The encoding of $\delta^{(2)}$ is distributed across both DMD1 and the SLM. Unlike the inputs x, the error vector is necessarily real-valued, and so requires both phase and amplitude encoding. However we must be able to rapidly update $\delta^{(2)}$ synchronously with x, requiring that the DMD alone is used to update the vector.

Device	Model	Function
Laser	Toptica DL100 ECDL	Generate forward and backward coherent beams
DMD	Texas Instruments DLP-6500 EVM	DMD1: encode and 'fan-out' input vector x DMD2: encode and 'fan-out' amplitude of error vector $\lvert\delta^{(2)}\rvert$
SLM1	Santec SLM-100	Signal region: encode weight matrix $W^{(1)}$ Phase-only region: encode sign of $\delta^{(2)}$
CL-1	Thorlabs 1" plano-convex round cylindrical lens, $f = 300$mm (LJ1558RM-B)	Perform fan-in for MVM1
Slit-1	Thorlabs V100	Select zero-order frequency component for MVM1
CCD1	Basler ace acA640-750um	Detect and measure MVM1 result, $a^{(1)}$
CCD3	Basler ace acA640-750um	Detect and measure MVM2b result, $\delta^{(1)}$
CL-2a	Thorlabs 1" plano-convex round cylindrical lens, $f = 300$mm (LJ1558RM-B)	Perform fan-out for MVM2a Perform fan-in for MVM2b
SLM2	Meadowlark E19x12-500-1200-HDMI	Encode weight matrix $W^{(2)}$
Slit-2	Thorlabs V100	Select zero-order frequency component for MVM2b
CL-2b	Thorlabs 1" plano-convex round cylindrical lens, $f = 300$mm (LJ1558RM-B)	Perform fan-in for MVM2a
CCD2	Ueye UI388xCP-M	Detect and measure MVM2a result, $z^{(2)}$

Table 4.1: List of devices and components used in experiment.

To achieve this, each vector element is encoded by two logical pixels on DMD2. The sign is encoded by the choice of logical pixel, 'left' or 'right', and the amplitude is encoded by the width of the chosen pixel. All the logical pixels are then mapped to SLM1, which has a fixed pattern of alternating phases. All regions that are mapped from the 'left' logical pixels have zero phase, so encode positive values; all regions mapped from the 'right' logical pixels have π phase, so encode negative values. Each pair of logical pixels are spatially separated, but this does not impact the optical

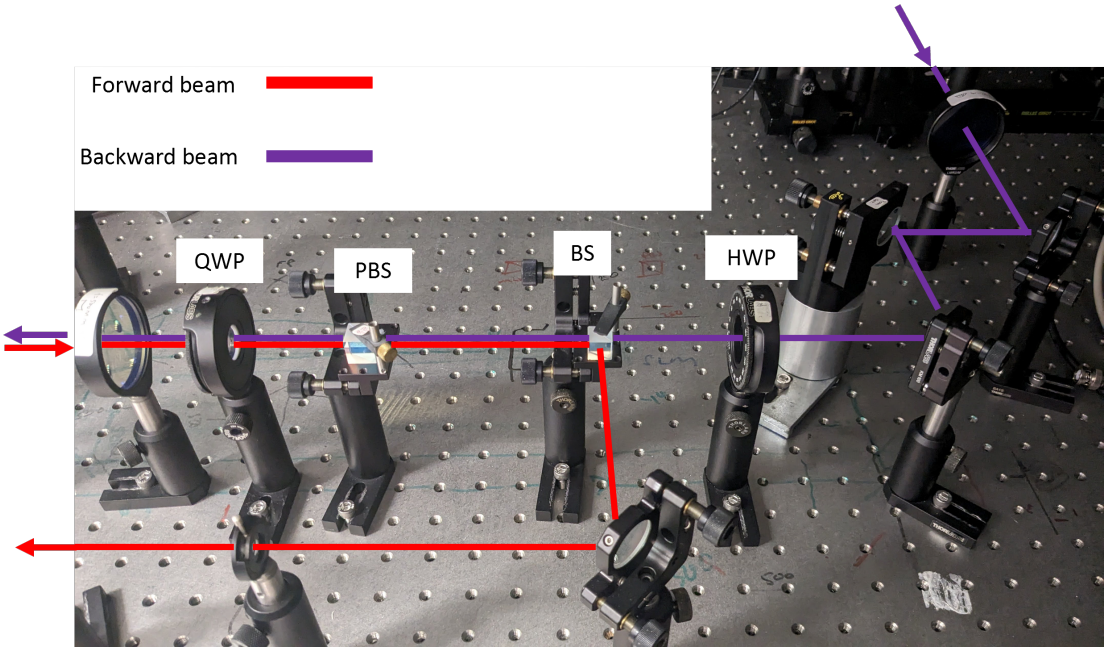


Figure 4.9: Photo from experiment to show how the backward beam is reintroduced to the system. QWP: quarter wave-plate; PBS: polarising beam-splitter; BS: beam-splitter; HWP: half wave-plate. QWP and PBS are used to remove back-reflections, as explained in the main text. HWP can be used to adjust backward beam power.

MVM scheme, as they are positioned such that they converge after passing through the cylindrical lens performing fan-in. Furthermore, the phase pattern on the SLM remains fixed, meaning the error vector can be updated entirely by the DMD.

After reflection from SLM1, the forward beam continues through the two-layer ONN, whilst the backward beam is separated by means of a pick-off mirror. Just as with the input vector, we utilise the full 2D plane of DMD2 to emulate the fan-out of error vector $\delta^{(2)}$, displaying M copies of the vector. This means we do not use lens set CL-2b to perform fan-out in the backward direction. Instead the backward beam was imaged with a $4f$ system directly from SLM1 to SLM2, via a beam splitter, and carefully aligned so as to overlap perfectly with the forwards beam. The optical system for reintroducing the backward beam is shown in Fig. 4.9.

SLM2 and narrow slits

A new model of LC-SLM, Meadowlark E19x12-500-1200-HDMI, was used for SLM2. The same calibration procedures as previously described were applied, although this modern device was found to have good phase linearity and uniformity, and

so required substantially less correction than the other models used. The same encoding method was also employed, identical to SLM1, using a diagonal blazed phase-grating and spatial filter to select the first order diffraction spot, allowing us to encode a real-valued weight matrix in the second layer. Crucially, this phase-grating method works in both forwards and backwards directions.

In anticipation of our scheme to perform optical nonlinear activation at the hidden layer, the positioning of the narrow slit was changed. In place of a single slit at the hidden layer plane, two slits were used. The locations of these slits are shown in Fig. 4.7, at the Fourier plane of the spatial filters after SLM1 (in the forwards direction) and after SLM2 (in the backwards direction). The action of the slits remains exactly the same, selecting only the zero-order frequency component in the horizontal direction. Ensuring the forwards and backwards beams passed cleanly through the second slit, and overlapped perfectly on SLM2, was sufficient to ensure the two beams were perfectly aligned.

Finally, different methods of coherent detection were required to measure the real-valued results of each of the MVMs. MVM2a, the second layer of the ONN in the forwards direction, was the most straightforward, following a similar scheme as previously. The cylindrical lens CL-2*b* acts to combine the signal with a phase-stable reference, generated by DMD1, which propagates throughout the ONN alongside (but spatially separate from) the signal. MVM1 and MVM2b involved performing two sequential measurements, with and without a weak phase-stable reference beam. The measurement without reference was used to find the amplitude, and comparing the intensities of the two measurements allowed the sign to be determined. Further details of these detection methods are detailed in Appendix C.2.

Eliminating back-reflections

With both forward and backward beams aligned and passing through the system, it was observed the beams would back-reflect from many of the optical components, and 'leak' into the opposite path. These back-reflections were clearly visible on CCD1 and CCD3, and would add a large error to the MVM results.

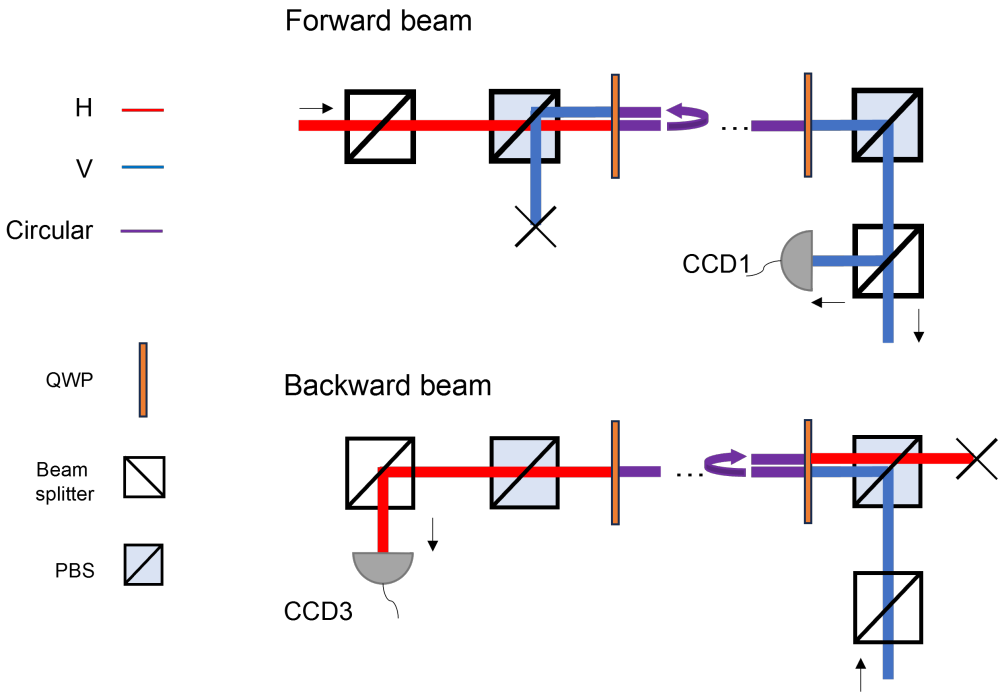


Figure 4.10: Conceptual scheme showing how back-reflections were prevented from interfering with both forward and backward signals. QWP: quarter wave-plate; PBS: polarising beam-splitter.

The solution was to introduce a series of quarter-waveplates (QWP) and polarising beam splitters (PBS) into the setup. Consider the simplified scheme in Fig. 4.10. The forward propagating beam is initially linearly polarised, say in the horizontal direction (H). The beam passes through the beam-splitter that is used to tap-off the backward beam. We then introduce a PBS and QWP, before propagating through the necessary optical components. The forward signal passes through the setup circularly-polarised, and any back-reflections must pass back through the QWP, becoming vertically polarised (V) and being fully removed by the PBS, so that they are not detected by CCD3.

Similarly, a QWP and PBS is introduced before the beam splitter used to tap-off the forward beam. A similar mechanism prevents back-reflections of the backward beam from mixing with the forward signal and being detected by CCD1.

4.2.3 Linear optical training - results

Cascaded MVMs

We begin by testing the performance of all three MVMs acting as a single optical system. We simultaneously perform the two-layer cascaded MVM1 and MVM2a, and the backwards MVM2b, by applying random weight matrices $W^{(1)}$ and $W^{(2)}$ to SLM1 and SLM2, encoding random vectors x and $\delta^{(2)}$ on DMD1 and DMD2, and simultaneously measuring the results $a^{(1)}$, $z^{(2)}$ and $\delta^{(1)}$. We test our system with small matrices, to find the precision in the best-case scenario; we use vectors and matrices with dimensions $N = 3$, $M = 5$ and $L = 2$. The positive-only input vectors are sampled randomly across $[0, 1]$, while the real-valued error vector elements, and both weight matrices, were sampled uniformly across $[-1, 1]$.

We show the results for 300 random simultaneous MVMs in Fig. 4.11. We plot the scatter of theory against measured values for each MVM, normalised by the maximum possible output when all elements are set to one. We see excellent agreement between experiment and theory, with all values falling along the diagonal line. The measured root mean squared errors (RMSEs) were 0.015, 0.008 and 0.035 respectively, giving SNR values of 14.9, 7.1 and 6.7, illustrated by the histograms plotted below the scatter, showing the distribution of noise in comparison to the signal.

MVM1 has good SNR consistent with previous experiments. The only exception is a few outliers close to zero. These values were calculated to have the incorrect sign, due to our new coherent detection method for this MVM. Further details are given in Appendix C.2. We show later how our optical activation function acts to almost entirely negate these errors.

In contrast, MVM2a has an SNR less than half that of MVM1. This is a result of noise accumulation from both layers, and the reduced signal range. This much smaller range of values for $z^{(2)}$ is expected, as a result of the random distribution of weights around zero. In order to estimate the level of noise inherent to MVM2a, independent of the first layer, we perform a simple simulation to analyse the noise

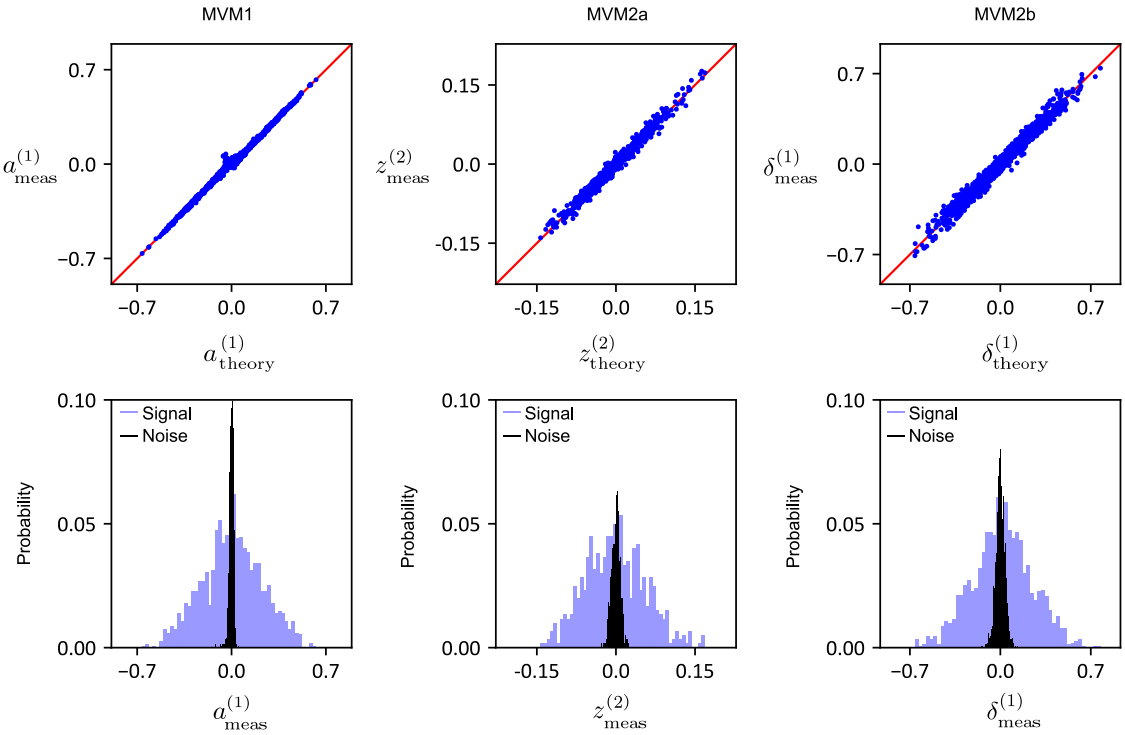


Figure 4.11: Results taken simultaneously for all three MVMs in the two-layer ONN with backpropagation. Scatter plots are shown of measured against theory results for MVM-1 (first layer forwards), MVM-2a (second layer forwards) and MVM-2b (second layer backwards), and associated histograms of the signal and noise error for each MVM are given underneath.

propagation. We simulate the cascaded MVM digitally, adding random Gaussian noise to each weight matrix, ζ_1 and ζ_2 respectively, so that

$$z^{(1)} = \frac{W^{(1)} \cdot x}{N} \tag{4.16}$$

$$z^{(2)} = \frac{W^{(2)} \cdot z^{(1)}}{M}, \tag{4.17}$$

and

$$z^{(1)}_{(\text{noise})} = \frac{\left(W^{(1)} + \zeta_1\right) \cdot x}{N} \tag{4.18}$$

$$z^{(2)}_{(\text{noise})} = \frac{\left(W^{(2)} + \zeta_2\right) \cdot z^{(1)}_{(\text{noise})}}{M}, \tag{4.19}$$

where we normalise each MVM result as we do in experiment, by the maximum possible result when all vector and matrix elements are unity i.e. by the dimensions of each input vector. We compare the error between the noiseless ground truth results, and simulated noisy results, for different values of ζ_1 and ζ_2, and empirically

	Training - forward	Training - backward	Inference
Digital simulator	Digital	Digital	Digital
Offline	Digital	Digital	Optical
Hybrid	Optical	Digital	Optical
Optical	Optical	Optical	Optical

Table 4.2: List of network training schemes showing which operations are performed digitally or optically.

search for the values that best match our experimental data. We find $\zeta_1 = 0.025$ and $\zeta_1 = 0.066$ give simulated SNR values of 14.9 and 7.4, closely matching our experiment results. We therefore conclude MVM1 has a noise level of 2.5%, and MVM2a has a slightly higher noise level of 6.6%.

Finally, in experiment MVM-2b has a lower SNR compared to MVM1, when they are functionally very similar MVMs. This is because the optical system is optimized for the forward direction: during the calibration procedures, the amplitude LUT was created using measurements from the forwards beam. Then since the same pattern is used on SLM2 to perform MVM2a and 2b simultaneously, there are likely to be non-uniform and non-linear responses for different elements in the backward beam. Furthermore, the optical elements were placed and carefully adjusted for the forward beam: including slit position and rotation, centering of the beam through the lenses, and most importantly the rotation of the cylindrical lenses in lens set CL-2a. Since the backward beam could only be adjusted with mirrors prior to being introduced to the system, there are aberrations and imprecision that cannot be adjusted for.

Optically-trained linear ONN

Before introducing the hidden-layer activation function, we tested the performance of the optical backpropagation scheme by training the linear two-layer ONN to recognise images from a reduced version of the MNIST dataset. The images were cropped and further downsampled to a size of 5×5, such that the input vector size is $N = 25$. We only classify three classes (digits '0', '1' and '2'), such that the output vector is $L = 3$, and through digital simulation we determined that ten hidden layer neurons were required for successful training, so $M = 10$. We used

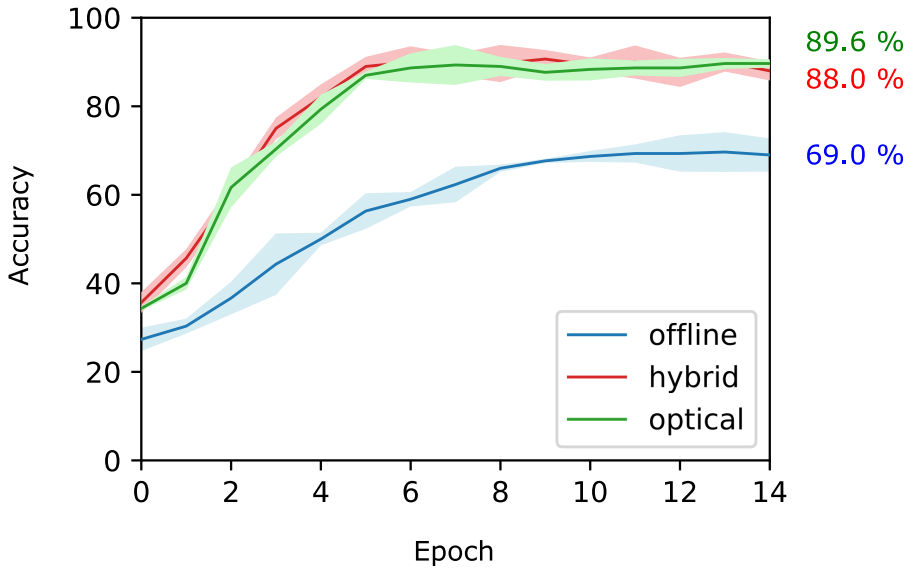


Figure 4.12: Learning curves of validation accuracy during the three types of network training performed with the two-layer linear ONN. For each training type, the mean and one standard deviation (over three repeat runs) of the validation accuracy measured after every epoch is plotted.

a small subset of the full MNIST dataset, with 400 training and 100 validation images. The training dataset was randomly split into 10 mini-batches of 40 images.

We first train a digital simulation of the ONN, with identical input data, layer sizes, hyperparameters and constraints, and use it to perform digital inference on the test dataset. The linear two-layer network is able to reach a maximum classification accuracy of 93%, the benchmark value to which we compare our optically-trained ONN.

We perform three types of training: 'offline', 'hybrid', and 'optical'. In all cases we use the ONN to perform inference on the validation set after each epoch. Offline training uses the digital simulator to calculate both the activations and errors, i.e. both forward and backward passes. Hybrid training uses the ONN for the forward pass only, and optical training implements our optical backpropagation scheme and performs both forward and backward passes with the ONN. We summarise these different training techniques in Tab. 4.2.

We plot the accuracy training curves for each ONN training method in Fig. 4.12. Specifically, we plot the mean and standard deviation of the accuracy measured for inference on the validation set, averaged over three repeats of the training with

different randomly sampled inputs in each mini-batch. We see both the hybrid and optical training significantly outperform offline training, and almost reach the benchmark accuracy of the digital simulator. Optical training achieves nearly 90%, outperforming hybrid training with 88%. This is a significant result that demonstrates the viability of our optical training scheme. These results indicate that optically training an ONN, at least at this relatively modest network size and depth, is entirely possible despite a significant level of noise in the linear layers.

We further analyse the optical training behaviour in Fig. 4.13 where we plot the evolution of all the neuron and error values for one optical training run. First, we separate every mini-batch based on class labels ('0', '1' and '2'). Then for each class, we individually plot the mean and standard deviation of each neuron in every mini-batch. We do this for every training iteration, i.e. for every weight update across all epochs (totalling 150 iterations), to visualise how the neuron values converge as the ONN evolves. We repeat this for the five critical vectors in our two-layer ONN. The first two rows plot the measured neurons at the hidden-layer, $a_j^{(1)}$ for $j \in [1, M]$, and at the output-layer, $z_k^{(2)}$ for $k \in [1, L]$. The middle row plots the network output, y_k for $k \in [1, L]$, after the softmax function has been digitally applied. Finally, the last two rows plot the measured error vectors, $\delta_k^{(2)}$ for $k \in [1, L]$ and $\delta_j^{(1)}$ for $j \in [1, M]$.

We can clearly see the training is successful in distinguishing the three classes, with a distinct output vector element $z_k^{(2)}$ (and similarly y_k) diverging to a large value for each class. Correspondingly, the output errors $\delta_k^{(2)}$ converge to zero as the training progresses. The backpropagated error $\delta_k^{(2)}$ clearly converges to zero for the '0' class, but this is less evident for the other classes. Similar behaviour was seen across all repeated optical training runs. This may indicate the network is relying on tuning the weights in the second layer to learn the required mapping of inputs to outputs. Therefore we analyse how these measured activation and error vectors impact the evolution of the weight matrix elements.

Recall the weight updates are calculated as

$$\Delta W_{ji}^{(1)} = \frac{\partial \mathcal{L}}{\partial W_{ji}^{(1)}} = x_i \cdot \delta_j^{(1)} \tag{4.20}$$

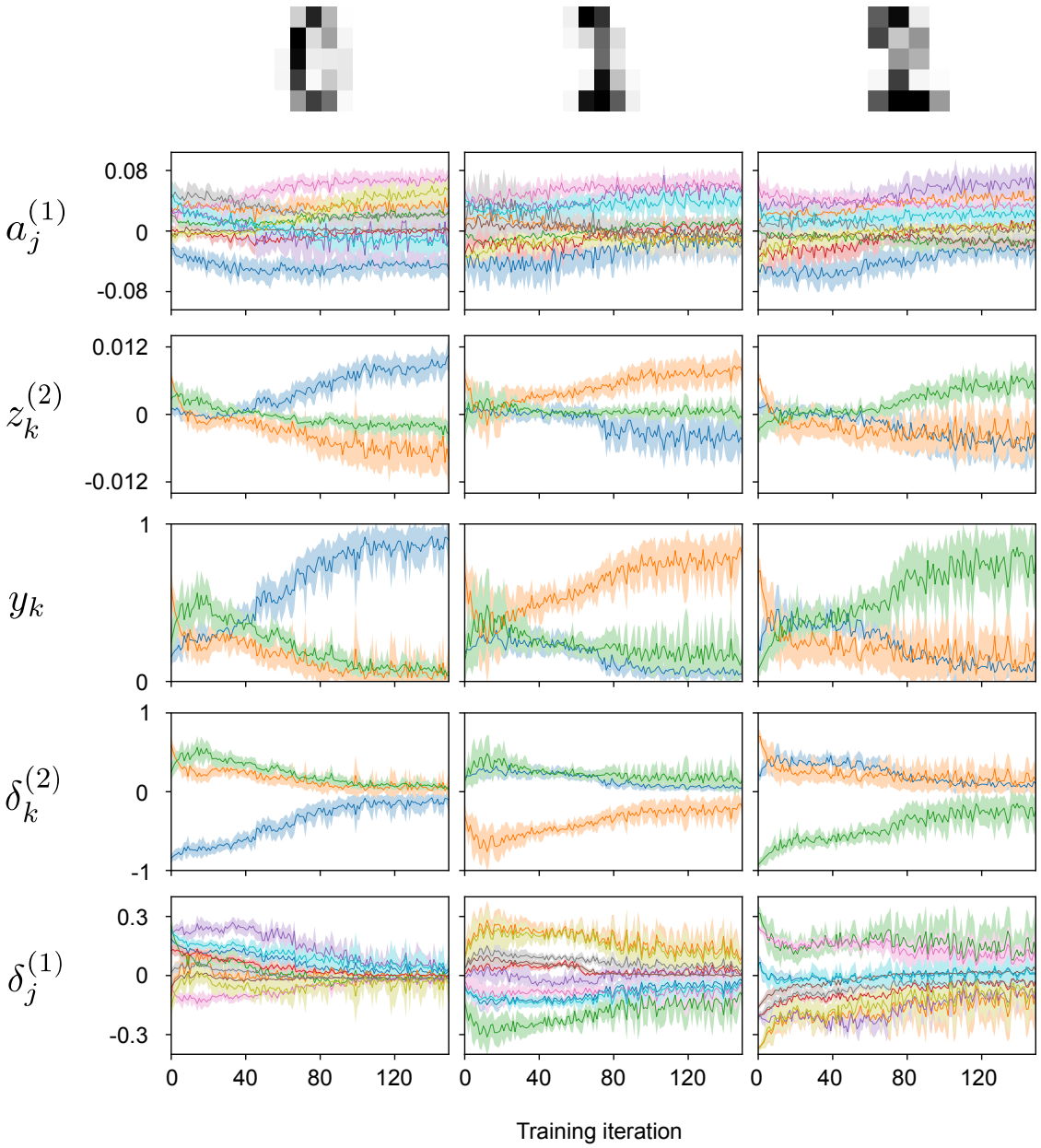


Figure 4.13: Evolution of forward activation and backward error values over the course of optical training. Plots are separated by class (MNIST digits 0, 1, 2). Each plot shows the mini-batch mean and standard deviation for every training iteration (i.e. every mini-batch across all epochs), for each vector element.

and

$$\Delta W_{kj}^{(2)} = \frac{\partial \mathcal{L}}{\partial W_{kj}^{(2)}} = a_j^{(1)} \cdot \delta_k^{(2)}. \tag{4.21}$$

We should expect to see the magnitude of these weight updates decrease over the course of training, as the weights converge towards their optimal values. In Fig. 4.14 we plot the evolution of the calculated weight updates, for both layers, during one

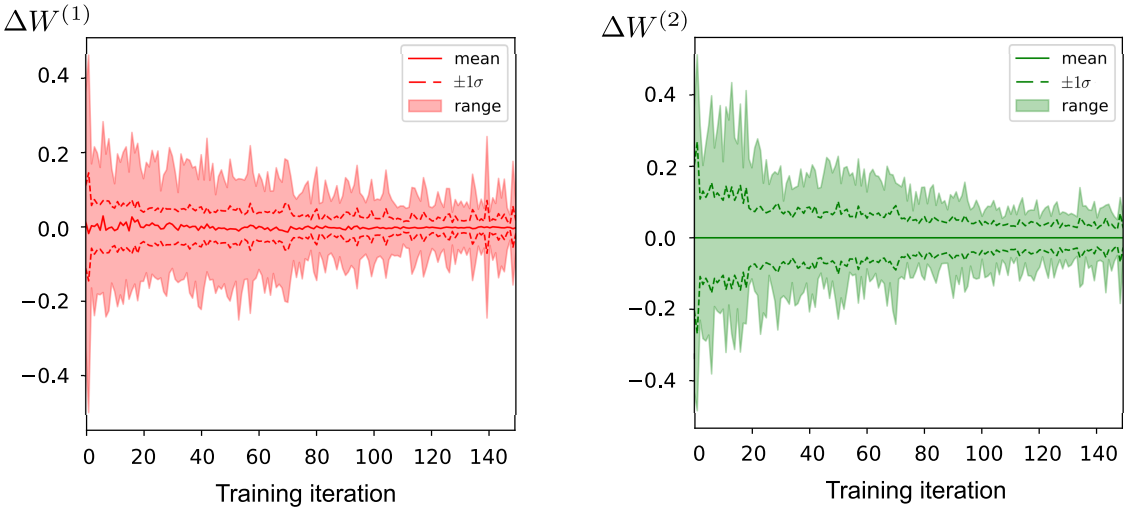


Figure 4.14: Evolution of weight updates for both layers during optical training. For each mini-batch, the maximum and minimum weight update value is plotted (range), as well as mean and one standard deviation.

optical training run. In particular we plot the range, mean and standard deviation across all the elements of each weight matrix, for every training iteration. We indeed see both matrix updates slowly converge towards zero, and conclude the training is not relying on tuning only one layer, which is crucial for the introduction of the nonlinear activation between the layers.

4.3 Conclusion

In this chapter we have described how the experiment setup was expanded beyond a single optical multiplier, creating a two-layer ONN that supports both forward and backward propagating beams. This ONN was formed by cascading two optical MVMs without opto-electronic conversion, and we explored how several experiment difficulties this raised were overcome.

The introduction of a narrow slit to select the zero-order frequency component after optical fan-in was particularly challenging. As noted in Chapter 1, performing optical fan-in can lead to unavoidable loss of optical power, and this manifests clearly in our experiment with the introduction of the narrow slit. The consequence was a reduction in the signal-to-noise ratio of the second MVM, with the degraded SNR measured in experiment inline with theory. Although this was not an issue in

this small-scale demonstration, optical loss and degradation of SNR through the network could be a bottleneck when trying to scale to larger and deeper ONNs.

We also introduced the principle of optical backpropagation, and how it was implemented in experiment by the introduction of a second beam propagating in the counter direction. Our hybrid training scheme was extended to perform optical training, where the activation vectors as well as the error vectors needed to calculate the necessary weight updates were calculated with optics, in every training iteration. The performance of both hybrid and optical training schemes against offline training were demonstrated with a simplified version of the MNIST classification task. Remarkably, optical training matched the classification accuracy, and number of iterations required for convergence, of hybrid training. This was an important demonstration to verify that optical training can converge to find the optimal parameters, despite the errors introduced to the calculated gradients by using analog hardware with finite SNR.

Next, in order to implement true end-to-end optical backpropagation, a suitable activation function performed by some optical phenomena must be introduced, that simultaneously supports both forward and backward propagating signals. We explore such an optical activation function in the next chapter.

5

Optical backpropagation through nonlinearity

Contents

Our two-layer ONN is no more expressive than a single-layer linear classifier unless a nonlinear activation function is introduced between the layers [89]. This chapter presents how we implement such an activation function in optics, without opto-electronic conversion, and with a scheme that allows optical backpropagation through the nonlinearity. Using this scheme, we were able to demonstrate for the first time the ability to train an ONN entirely with optics. Some of the results and figures in this chapter are presented in the preprint '*Training neural networks with end-to-end optical backpropagation*' [160](Author contribution: joint-first author, carried out experiments and performed data analysis. Equal contribution to

manuscript preparation.)

5.1 Saturable absorption activation function

5.1.1 Concept

We use exactly the same experiment setup as before, but now apply an activation function to the result of MVM1 before continuing to MVM2a:

$$a_j^{(1)} = g\left(z_j^{(1)}\right), \tag{5.1}$$

where $g(\cdot)$ is the hidden-layer activation function applied element-wise to the result of MVM1, $z^{(1)}$.

To implement this activation function in optics, and maintain the ability to perform optical backpropagation, we must choose some optical phenomenon that satisfies three conditions:

1. Has a nonlinear response in the forwards direction.
2. Has a linear response in the backwards direction.
3. Modulates the backward light by the derivative of the forward nonlinear function.

Simultaneously satisfying the first two requirements seems challenging, let alone the much stricter requirement of the third, since most optical media exhibit similar properties for forward and backward propagation. However it has often been observed that many optical media exhibit nonlinear properties for strong optical fields, but are approximately linear for weak fields. We can therefore satisfy the conditions (1) and (2) by introducing the backward beam with a much lower intensity than the forward beam.

The third requirement is a result of the gradient calculations in the backpropagation algorithm. Recall the calculation to find the error vector at the first layer is

$$\delta_j^{(1)} = \rho_j^{(2)} \cdot g'\left(z_j^{(1)}\right), \tag{5.2}$$

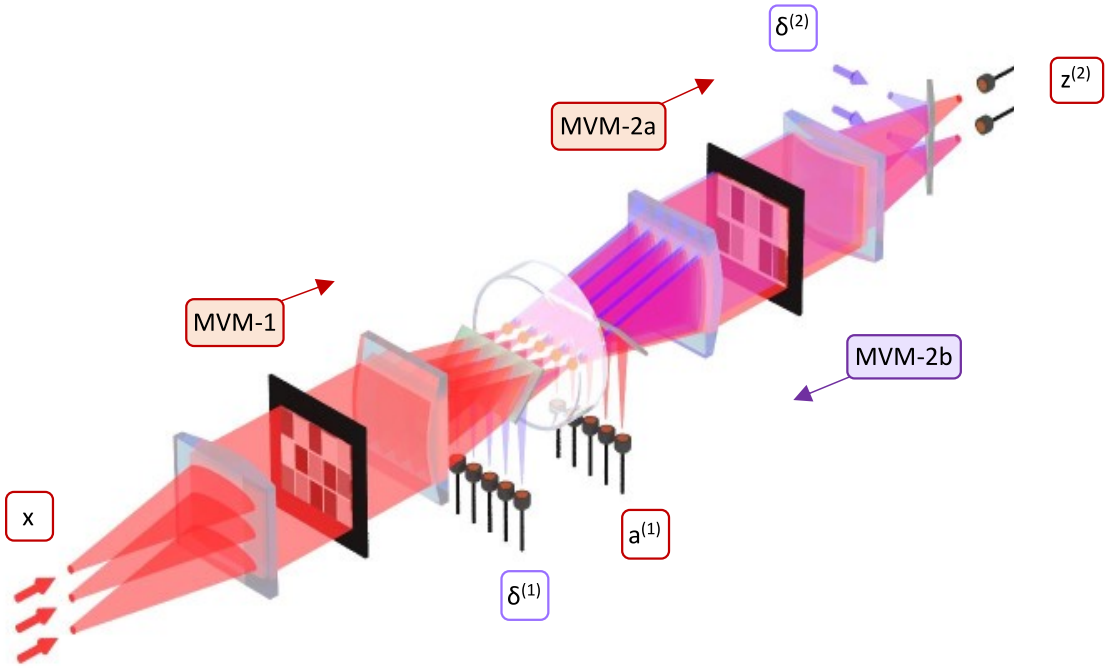


Figure 5.1: Conceptual diagram for two-layer ONN with backpropagation and saturable absorption nonlinearity at the hidden layer.

where

$$\rho_j^{(2)} = \sum_k W_{jk}^{(2)} \delta_k^{(2)}. \tag{5.3}$$

After calculating the term $\rho_j^{(2)}$ with MVM2b exactly as before, the resultant vector must now be element-wise multiplied by the new term $g'\left(z_j^{(1)}\right)$. Note that this must happen physically at exactly the same position in the optical setup as the nonlinear function $g\left(z_j^{(1)}\right)$ is being implemented for the forward beam. We can therefore look to use some optical process that allows the forward and backward beams to influence one-another, to achieve the desired transmission response in both directions.

Previous theoretical work by other members of our lab group discovered just such an optical process that satisfies the conditions to a surprisingly high level of precision. Extensive details, including simulations for using the scheme in optical training, have been previously presented [143], and the theory is discussed in the next section. The key idea is to use the process of saturable absorption (SA), whereby the degree of absorption through a medium is reduced at high input optical intensity. Many materials demonstrate this property, and the effect is used across many applications,

including doppler-free spectroscopy [161, 162], and mode locking and Q-switching of lasers [163]. This scheme also works for the process of saturable gain, which would give the added benefit of optical amplification between layers of an ONN.

In our experiment we utilise the phenomenon of SA in a vapor cell of Rubidium atoms when the laser is tuned to resonance with an atomic transition. A conceptual scheme of the complete two-layer ONN, including the vapor cell used to implement the SA activation function, is shown in Fig. 5.1. The forwards and backwards beams overlap perfectly within the cell, and their combined interaction with the atoms determines their transmission, as explored below.

5.1.2 Theory

Consider a thin absorbing material, and incident light with a weak input intensity I_{in}. The transmitted intensity I_{out} is given by the Beer-Lambert law

$$I_{\text{out}} = I_{\text{in}} e^{-\alpha_0}, \tag{5.4}$$

where α_0 is the small-signal resonant optical depth of the material. We model such a material as a collection of simple two-level atoms, and assume the beam is tuned perfectly on resonance with the atomic transition between the ground and excited state. A sufficiently weak beam (significantly below the threshold intensity) experiences linear absorption as it excites the transition, and shows a linear reduction in transmission. Meanwhile for a strong enough beam (at or above the threshold intensity) the absorption is saturated as the ground state population is depleted, and shows a distinct nonlinear transmission.

The degree of absorption is therefore dependent on the intensity of the incident beam, so the coefficient α becomes dependent on I, according to

$$\alpha(I) = \frac{\alpha_0}{1 + \frac{I}{I_s}}, \tag{5.5}$$

where I_s is the saturation intensity of the material. For a thin cell, we can approximate the new transmission as

$$I_{\text{out}} = I_{\text{in}} e^{-\alpha(I_{\text{in}})} \tag{5.6}$$

$$= I_{\text{in}} \exp\left(-\frac{\alpha_0}{1+\frac{I_{\text{in}}}{I_s}}\right), \tag{5.7}$$

where we have neglected varying intensity within the cell. Note the rigorous derivation without making such an approximation gives a transcendental equation, see for example [164].

Equivalently, we can say the electric field amplitude changes according to

$$E_{\text{out}} = E_{\text{in}} \exp\left(-\frac{\alpha_0/2}{1+\left[\frac{E_{\text{in}}}{E_s}\right]^2}\right) \tag{5.8}$$

$$= g(E_{\text{in}}), \tag{5.9}$$

where E_s is the electric field saturation threshold, and we refer to $g(\cdot)$ as the saturable absorption (SA) nonlinearity function.

We now consider a setup with two counter-propagating beams passing through the cell, one 'pump' beam with intensity many times above the threshold intensity, and one much weaker 'probe' beam, below the threshold. This is a standard and common-place setup used in Doppler-free spectroscopy experiments. The transmission of the pump beam will be as given in (5.9), highly nonlinear around the saturation intensity:

$$E_{\text{out}}^{(\text{pump})} = g\left(E_{\text{in}}^{(\text{pump})}\right). \tag{5.10}$$

In contrast, the weaker probe beam cannot change the degree of transmissivity, and will experience only linear absorption. Crucially, the absorption coefficient for the probe will depend entirely on the intensity of the pump. Mathematically,

$$E_{\text{out}}^{(\text{probe})} = E_{\text{in}}^{(\text{probe})} e^{-\alpha\left(I_{\text{in}}^{(\text{pump})}\right)/2} \tag{5.11}$$

$$= E_{\text{in}}^{(\text{probe})} \exp\left(-\frac{\alpha_0/2}{1+\left[\frac{E_{\text{P}}}{E_s}\right]^2}\right), \tag{5.12}$$

where we use shorthand notation $E_{\mathrm{P}} \equiv E_{\mathrm{in}}^{(\mathrm{pump})}$. This pump-probe process exactly matches the requirements of the nonlinear activation. We take our forward propagating beam as the pump, and the backward propagating beam as the much weaker probe. We have satisfied conditions (1) and (2), with a nonlinear response $g(\cdot)$ acting on the forward beam, and a linear response for the backward beam. We also have the necessary conditions for (3), with the linear response on the backward beam dependent on the intensity of the forward beam.

We can imagine two limiting cases to see how this works. If the pump is infinitely strong, the atoms will completely saturate, and the backward probe will not be absorbed. The probe transmission will be 100%, effectively multiplying the probe value by unity. In contrast, if the pump is turned off, no atoms are excited, and since the probe is kept at a brightness far below the saturation intensity, it will be entirely absorbed. The probe transmission will be zero, effectively multiplying the probe value by zero.

What remains is to check this pump-probe dependence is a good approximation for the derivative of $g(\cdot)$ in between these two extremes. This derivative can be found directly from Eq. (5.9), giving

$$g'(E_{\mathrm{P}}) = \beta\left(E_{\mathrm{P}}\right) \exp\left(-\frac{\alpha_0/2}{1+\left[\frac{E_{\mathrm{P}}}{E_s}\right]^2}\right), \tag{5.13}$$

where

$$\beta\left(E_{\mathrm{P}}\right) = 1 + \frac{\alpha_0 \cdot \left[\frac{E_{\mathrm{P}}}{E_s}\right]^2}{\left(1+\left[\frac{E_{\mathrm{P}}}{E_s}\right]^2\right)^2}. \tag{5.14}$$

This means we can rewrite Eq. (5.12) as

$$E_{\mathrm{out}}^{(\mathrm{probe})} = E_{\mathrm{in}}^{(\mathrm{probe})} g'\left(E_{\mathrm{in}}^{(\mathrm{pump})}\right) \cdot \frac{1}{\beta\left(E_{\mathrm{P}}\right)}. \tag{5.15}$$

This is precisely the required response of the probe, being linearly modulated by the derivative of the nonlinear function on the pump, up to the factor β. What Guo et al. showed was that this factor can be considered approximately constant over the nonlinear region, and that the additional linear scaling factor

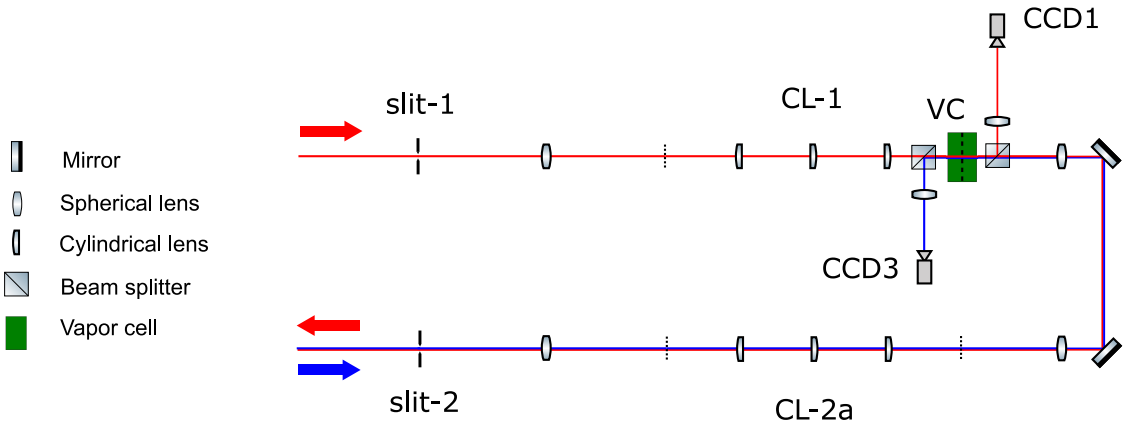


Figure 5.2: Diagram of experiment setup around the vapor cell.

can then be safely ignored in the optical training scheme (as it can be factored into the chosen learning rate).

Therefore the pump-probe scheme with SA nonlinearity is an ideal scheme to implement optical backpropagation, and we associate the ONN vectors with the pump and probe fields:

$$z^{(1)} \equiv E_{\text{in}}^{(\text{pump})} \tag{5.16}$$

$$a^{(1)} = g\left(z^{(1)}\right) \equiv E_{\text{out}}^{(\text{pump})} \tag{5.17}$$

$$\rho^{(2)} \equiv E_{\text{in}}^{(\text{probe})} \tag{5.18}$$

$$\delta^{(1)} = \rho^{(2)} \cdot g'\left(z^{(1)}\right) \equiv E_{\text{out}}^{(\text{probe})}. \tag{5.19}$$

5.1.3 Methods

Experiment setup

In our experiment we implement the SA nonlinearity using a heated vapor cell of Rubidium atoms. A $1''$ diameter vapor cell is placed at the output plane of MVM1, as shown in Fig. 5.2, and kept at $70°C$ using a heating jacket and PID temperature controller (Thorlabs TC200). Fig. 5.3 shows photos from experiment of the vapor cell, surrounded by the heating jacket. The second image is taken through an IR viewer, allowing the laser beam to be seen as it passes through the cell and fluoresces.

The forward beam that passes through the cell is the narrow vertical beam encoding the result of MVM1, $z^{(1)}$, with slit-1 having already selected the zero-order

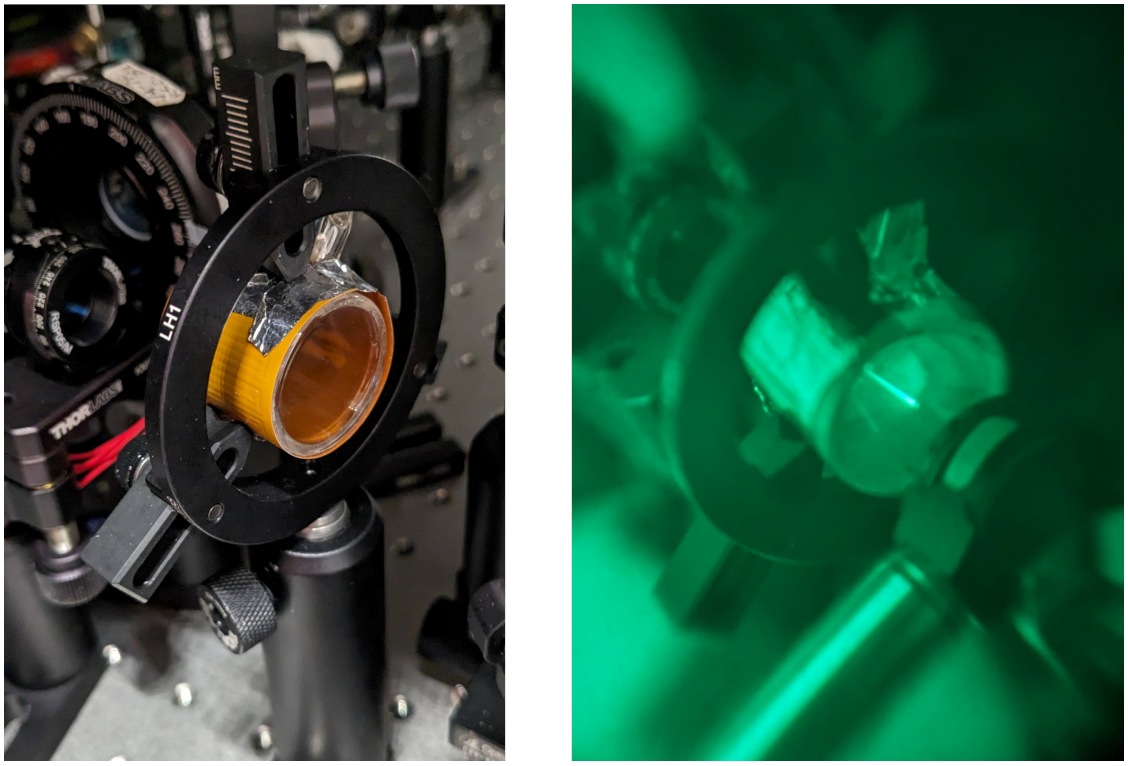


Figure 5.3: Photos of vapor cell used in experiment. Left image shows the simple heating jacket used to maintain cell temperature. Right image is taken through an infrared viewer, showing beam fluorescence.

frequency component. Each vector element, or neuron, of the hidden-layer vector $z_j^{(1)}$ is represented by an optical mode passing through the vapor cell, which each independently experience a nonlinear transmission response dependent on their intensity, according to Eq. (5.10). The backward beam that passes through the cell encodes the results of MVM-2b, $\rho^{(2)}$. Slit-2 has selected the zero-order frequency component of this vector, and each optical mode of the backward beam, representing $\rho_j^{(2)}$ overlaps perfectly with the corresponding forward optical mode $z_j^{(1)}$.

Beam splitters before and after the vapor cell tap-off a portion of the beams, and the intensity of each beam is measured with CCD1 and CCD3, exactly as before.

Laser wavelength

The laser is tuned to resonance with the D_2 atomic transition close to 780nm. A portion of the laser was tapped-off and used to perform Doppler-free pump-probe spectroscopy on a second Rubidium vapor cell. This allowed the multiple transition

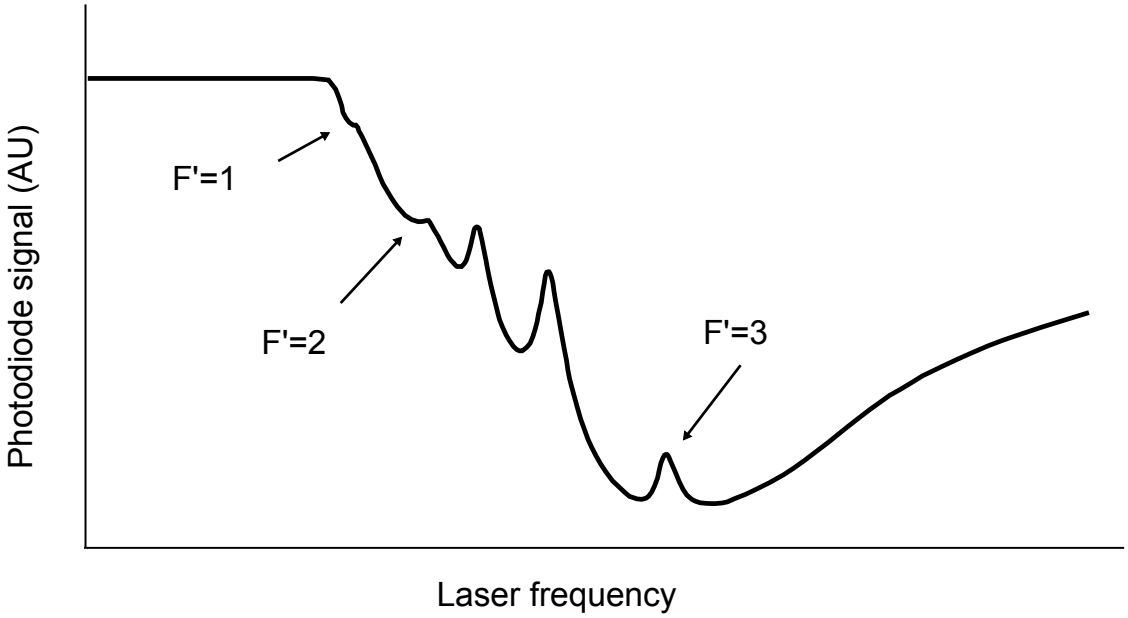


Figure 5.4: Recorded transmission spectrum of rubidium vapor cell used to lock laser wavelength. Horizontal axis represents laser frequency, modulated by scanning laser cavity length. The hyperfine transition peaks from $F = 2$ to $F' = 1, 2, 3$ are indicated.

peaks resulting from the hyperfine structure of Rubidium to be resolved, which would otherwise be lost due to Doppler broadening. Both cells contained the Rb^{87} isotope, and the measured transmission spectrum for the $5^2S_{1/2} \rightarrow 5^2P_{3/2}$ transition is shown in Fig. 5.4, with the hyperfine transition peaks from $F = 2$ to $F' = 1, 2, 3$ indicated. Using a custom-built dither-locking apparatus, the laser wavelength was locked to the $F' = 3$ peak. This was chosen for having a distinct, symmetric peak and was found to have the largest optical depth. Further details of the locking procedure are provided in Appendix D. Attempts at performing the experiment without locking the laser wavelength were unsuccessful, as the wavelength was liable to drift over the course of optical training. As the wavelength deviated away from resonance with the transition, the effective optical depth of the cell decreased, and the training faltered as the nonlinear activation function changed.

Optical power

The beam was required to be sufficiently powerful such that each optical mode could locally saturate the cell. This power depends on the saturation intensity of the atoms, and the beam area of each spot. For the $F = 2 \rightarrow F' = 3$ transition of

Rb^{87} the saturation intensity is theoretically given as $I_{\text{sat}} = 3.576(4)$ mW/cm^2 [165], and the area of each spot was approximately 60μm $\times 120\mu$m: the width was determined by the chosen slit width, which as explained in Sec. 4.1.2 was $w_{\text{slit}} = 60\mu$m, and the vertical height was limited by crosstalk between the optical modes; neighbouring modes were liable to influence the transmission of one another, as they were able to partially saturate surrounding atoms. This required a minimum separation of modes, and given the finite size of the beam within the cell, limited the possible number of hidden layer neurons. It was empirically found that using $M = 5$ neurons could be accommodated with negligible crosstalk, with a height approximately $2w_{\text{slit}} = 120\mu$m.

Therefore, the required power per optical mode to reach the saturation intensity is at minimum $P_{\text{sat}}^{(\text{theory})} = 0.25\mu$W. This theoretical value is however only a crude approximation, and there are many factors that may act to change the saturation intensity:

- increased effective spot size due to divergence of the beam during propagation through the vapor cell,
- broadening mechanisms, including Doppler, pressure, and transit-time broadening,
- non-saturable absorption mechanisms,
- buffer gas in the vapor cell.

It was found in experiment that each optical mode actually required $P_{\text{sat}}^{(\text{meas})} = 3.0\mu$W to reach saturation. We explain how this number was found empirically below, and a similarly large saturation power, as compared to the literature value, was consistently found across many tests with the available vapor cell. The exact value of the saturation intensity is inconsequential for our experiment so long as sufficient power can be provided to saturate the cell; for successful optical training, the range of values typically taken by each neuron, $z_j^{(1)} \in [0, z_{\max}]$, must be represented by a range of intensities many times the saturation intensity.

The maximum power being used for the forward beam was 140mW. In the experiment, there are many sources of loss in the optical system:

- To achieve a uniform input intensity profile, the input Gaussian spot onto the DMD was broadened, and only a small central part of the beam was used, approximately 10% of the power,
- the DMD had very poor reflectivity (likely due to the coating not matching the laser wavelength) and significant diffraction from the pixelated structure, so had an efficiency of only 30%,
- the LC-SLM encoding method of a blazed phase grating, combined with imperfect reflection, has an efficiency of less than 50%,
- the narrow slit further reduces the optical power by approximately 50%.

In total, the system efficiency from input beam to vapor cell was only 0.3%, such that a total of 420μW was incident on the cell. With $M = 5$ modes, each mode therefore had a maximum power of 84μW, so that

$$I_j^{(\text{pump})} \in [0, 27.7 I_{\text{sat}}]. \tag{5.20}$$

From simulation, this is adequate for the SA nonlinearity to be suitable for optical training.

The lifetime of the Rb^{87} $5^2S_{1/2} \to 5^2P_{3/2}$ transition is $\tau = 26$ ns [165]. In order to saturate the cell, the optical pulse of each forward pass through the network must be longer than this lifetime, and we can therefore estimate each neuron requires at least $P_{\text{sat}}^{(\text{meas})} \cdot \tau = 80$ fJ for the nonlinear activation. Similarly, the lifetime places an upper limit of approximately 40 MHz on the clock rate of an ONN using this mechanism, although in this work the limiting factor is the CCD camera operating in kHz range.

Probe measurement

The optical training scheme requires the backward-propagating error beam, referred to throughout as the probe beam, to undergo linear absorption only. As motivated above, to achieve this the probe intensity must be less than the saturation intensity of the cell, however the probe must also have sufficient intensity to be detected by CCD3 with a good SNR. In experiment, we set each optical mode of the probe, representing one element of the vector $\rho_j^{(2)}$, to have a maximum intensity approximately 2% of the maximum pump, at the vapor cell. Therefore we have

$$I_j^{(\text{probe})} \in [0, 0.6I_{\text{sat}}]. \tag{5.21}$$

The symmetry of our experiment setup meant the width of each probe optical mode was identical to the corresponding pump mode, determined by w_{slit}. The height of each probe mode was chosen to be slightly smaller than the pump, approximately $1.5w_{\text{slit}} = 90\mu$m, to ensure the probe modes were completely overlapped by the pump.

There are two notable experimental deviations from the theory outlined above. First, the probe is not fully absorbed by the cell even with the pump turned off. In practice we found about 10% of the probe power was transmitted regardless of the pump intensity. This corresponds to the probe experiencing a much lower optical depth, approximately $\alpha_0 = 2$, than was measured by analysing pump transmission. Second, a strong pump caused the atoms to fluoresce in all directions, including along the backward probe path. This created a background offset to the probe, proportional to the forward pump signal.

In order for the probe measurement to approximate the gradient of the pump response, we must negate these two background terms. This was achieved by taking multiple intensity measurements to determine $\delta^{(1)}$: pump-only, probe-only and both (pump-probe).

First, in each training iteration we perform the forward pass through the network to measure the activations. At the same time we measure the pump leakage into the backward path with camera CCD3. This gives the background term due to the pump fluorescence, I_{P}.

Second, after calculating the loss and $\delta^{(2)}$, the backward beam is passed through the system with the pump turned off, and the second background term I_{pr} is measured. This accounts for transmission of the probe through the cell that is independent of the pump, and the lower effective optical depth of the probe.

Next, in order to measure the sign of the result of MVM-2b, $\rho^{(2)}$, a bright reference beam is encoded in parallel with the error vector $\delta^{(2)}$ (using the DMD and SLM1) and both interfere after summation with the cylindrical lens. The intensity $I_{\text{pr+ref}}$ is measured, and can be used to measure the sign of the real-valued vector $\rho^{(2)}$, with exactly the same single-shot coherent detection method as MVM-2a.

Finally, both forward and backward beams are turned on (without the reference beam). The pump-probe intensity I_{both} is captured by CCD3, and all measurements are combined to calculate $\delta^{(2)}$ as

$$\delta^{(1)} = \text{sign}\left(\rho^{(2)}\right) \cdot \left(\sqrt{I_{\text{both}}} - \sqrt{I_{\text{P}}} - \sqrt{I_{\text{pr}}}\right). \tag{5.22}$$

5.1.4 Results

Pump response

We first test the nonlinear response of the forward pump beam. For each of the $M = 5$ modes, labelled by $j \in [0, M-1]$, we far de-tune the laser away from resonance and scan the input field amplitude, $E_{\text{in}}^{(\text{pump})}$ using SLM1. When the DMD and SLM1 are set to maximum reflection, and the laser is far de-tuned from resonance, we define the field amplitude at the cell as E_{max}, and use this to normalise all field amplitudes. The output intensity, $I_{\text{out}}^{(\text{pump})}$ is measured (in arbitrary units) by CCD1. To normalise across all five modes, the actual power of each mode at maximum brightness was also recorded with a power meter (Thorlabs PM100D), and each mode scaled appropriately to normalise by the global maximum, defined as P_{max}, with corresponding intensity I_{max}. The output field amplitude is calculated as

$$E_{\text{out}}^{(\text{pump})} = \sqrt{\frac{I_{\text{out}}^{(\text{pump})}}{I_{\text{max}}}} E_{\text{max}}. \tag{5.23}$$

Figure 5.5 shows the linear response of each optical mode when the laser is tuned off-resonance (red crosses) is almost perfect, as expected from the careful

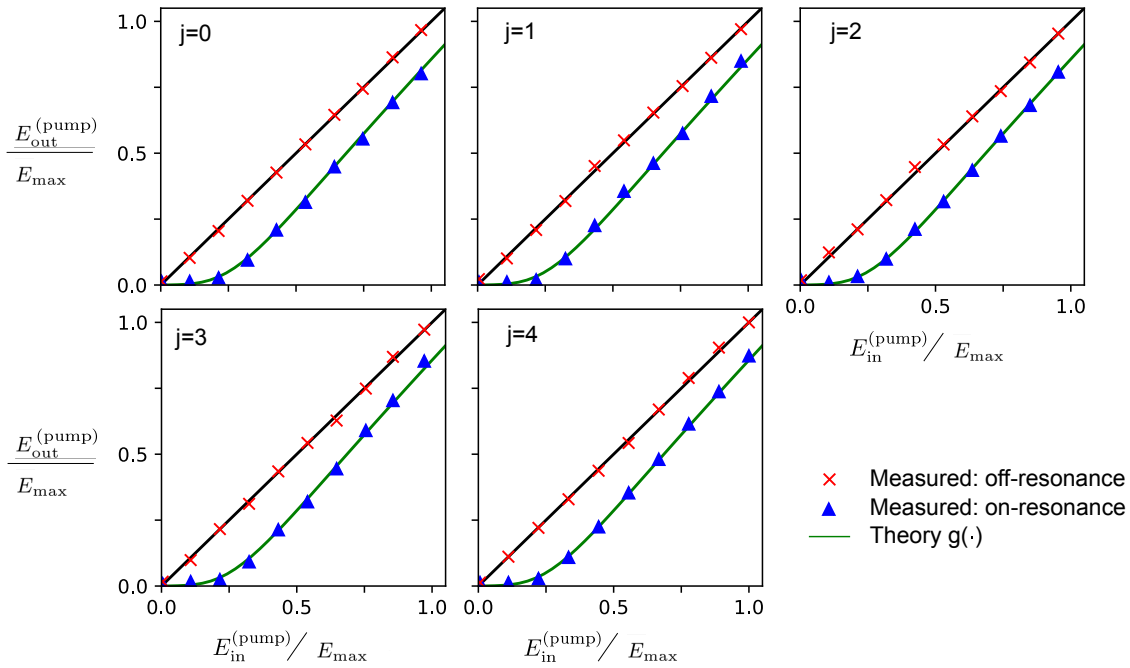


Figure 5.5: Measured response of the pump transmission as a function of pump input, when the laser is tuned on-resonance (blue triangle) and off-resonance (red cross). The response of each optical mode $j \in [0, 4]$ is shown. The theoretical SA response is also shown (green curve). All values are normalised by $E_{\max}$, which is defined relative to $P_{\max} = 83\ \mu$W, as given by Eq. (5.23) in the main text.

calibration of the DMD and SLM1. The test is repeated, with the laser tuned and locked to resonance. Figure 5.5 shows the nonlinear response (blue triangles) matches extremely well to our simple theoretical model. Note the same value of $I_{\max}$ is used for normalisation in both cases. We numerically fit the SA function, as given by (5.10), with the combined dataset of all five modes, leaving the optical depth α_0 and saturation field amplitude E_{sat} as free parameters. We find the values $\alpha_0 = 8.6$ and $E_{\text{sat}} = 0.19 E_{\max}$. The maximum power measured was $P_{\max} = 83\ \mu$W, and together these measurements were used to calculate the saturation power per optical mode $P_{\text{sat}}^{(\text{meas})} = 3.0\ \mu$W.

Probe response

We next test the probe response. For the backward probe beam the transmission of each optical mode is a function of two variables: the probe input intensity, and the pump intensity. We tune and lock the laser to resonance, and for each of the $M = 5$ modes, labelled by $j \in [0, M-1]$, perform two measurements. First, we fix

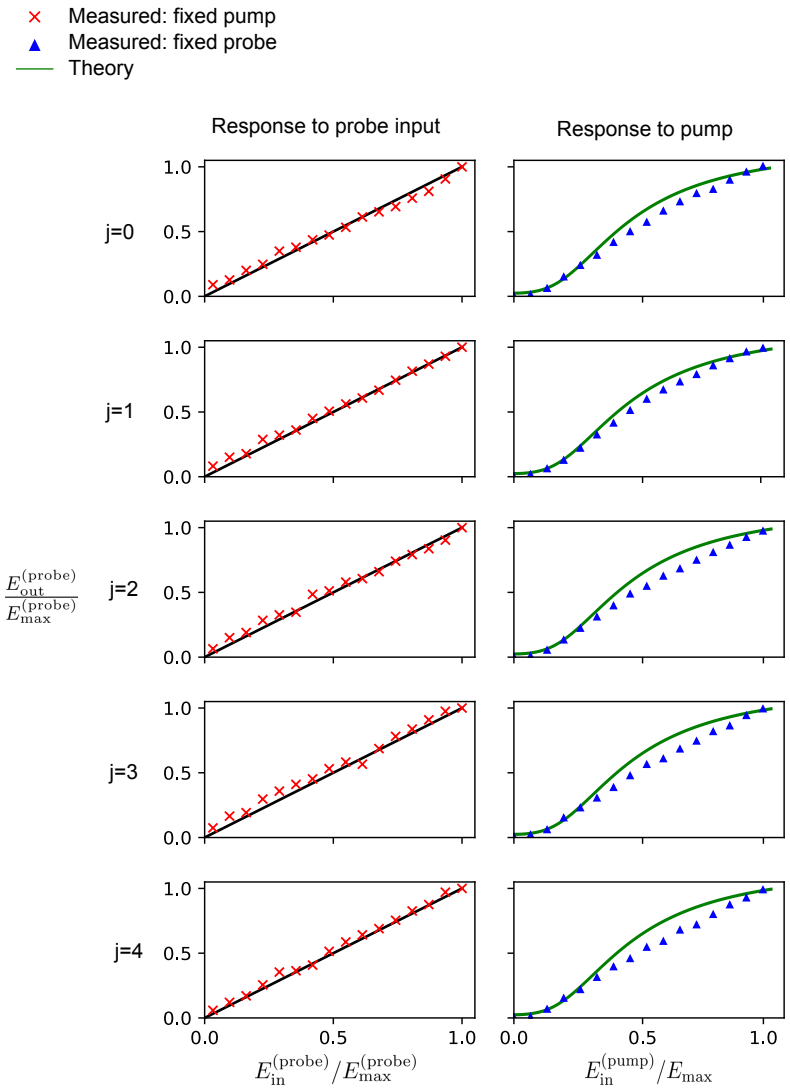


Figure 5.6: Measured response of the probe transmission as a function of probe input with fixed pump (red crosses), and as a function of pump input with fixed probe (blue triangles). The measured probe response is calculated from multiple measurements to correct for leakage of pump and unabsorbed probe. The theoretical response shown (green curve) uses the parameters found from the previous pump response experiment. All values are normalised appropriately to lie in the range $[0, 1]$: pump values are normalised by E_{max}, which is defined in the main text relative to $P_{\text{max}} = 83\ \mu\text{W}$; probe values are normalised by $E_{\text{max}}^{(\text{probe})}$, which is defined in the main text relative to $P_{\text{max}}^{(\text{probe})} = 1.6\ \mu\text{W}$.

the pump at I_{max}, fully saturating the vapor cell. We then scan the probe input electric field amplitude, $E_{\text{in}}^{(\text{probe})}$, using SLM2. We measure the output intensity (in arbitrary units) using camera CCD3, and calculate the output field amplitude $E_{\text{out}}^{(\text{probe})}$ (red crosses) by taking the square root and normalising to the range $[0, 1]$ independently for each mode. Figure 5.6 shows the response is close to being linear,

but is not as precise as the pump. This is primarily due to imperfect calibration with SLM2, which recall is optimised for the forward beam in MVM-2a.

Second, we fix the probe input at maximum intensity, and scan the pump field amplitude, $E_{\text{in}}^{(\text{pump})} \in [0, E_{\max}]$, using SLM1. We should expect the probe transmission to change according to (5.12):

$$E_{\text{out}}^{(\text{probe})} = E_{\text{in}}^{(\text{probe})} \cdot \exp\left(-\frac{\alpha_0/2}{1 + \left[\frac{E_{\text{in}}^{(\text{pump})}}{E_{\text{sat}}}\right]^2} \right). \tag{5.24}$$

We define the field amplitude of the probe output, when pump and probe inputs are maximum, to be $E_{\max}^{(\text{probe})}$, and normalise by this value throughout, such that all values fall in the range $[0, 1]$. Substituting the same values of optical depth $\alpha_0 = 8.6$ and saturation field $E_{\text{sat}} = 0.19 E_{\max}$ as measured for the pump response, we can rewrite (5.24) as

$$\frac{E_{\text{out}}^{(\text{probe})}}{E_{\max}^{(\text{probe})}} = 1.16 \ \exp\left(-\frac{\alpha_0/2}{1 + \left[\frac{E_{\text{in}}^{(\text{pump})}}{E_{\text{sat}}}\right]^2} \right). \tag{5.25}$$

We plot this normalised theoretical response in Fig. 5.6 (green curve). We then plot the measured probe response (blue triangles), calculated using the multiple-measurement scheme to correct for the probe offset and pump fluorescence:

$$\frac{E_{\text{out}}^{(\text{probe})}}{E_{\max}^{(\text{probe})}} = \frac{1}{\sqrt{I_{\max}^{(\text{probe})}}} \left(\sqrt{I_{\text{both}}} - \sqrt{I_{\text{pump}}} - \sqrt{I_{\text{probe}}} \right), \tag{5.26}$$

where $I_{\max}^{(\text{probe})}$ is the intensity of the globally measured brightest probe output, which was measured to have power $P_{\max}^{(\text{probe})} = 1.6\ \mu\text{W}$.

The measured response does not fit perfectly to the theory, but this is expected given the simplicity of the model used, the number of phenomena and additional mechanisms unaccounted for, and that the values of α_0 and E_{sat} were calculated in the previous experiment from the pump response only. However the model does appear to be accurate at smaller pump intensity, and all five neurons are consistent.

Pump-probe response with MVM

Finally, we test the simultaneous pump and probe responses when random vectors and matrix elements are encoded on the DMD and SLM1, performing MVM1 so that all pump neurons take on random values at the cell simultaneously, whilst keeping the probe at uniform maximum brightness. We use the same random input vector and weight matrices as for the results presented in Sec. 4.2.3. We associate the properly normalised field amplitudes with the theory and measured ONN vectors as

$$E_{\text{out}}^{(\text{pump})} \equiv a_{\text{meas}}^{(1)}, \tag{5.27}$$

$$E_{\text{out}}^{(\text{probe})} \equiv \delta_{\text{meas}}^{(1)}, \tag{5.28}$$

$$E_{\text{in}}^{(\text{pump})} \equiv z_{\text{theory}}^{(1)}, \tag{5.29}$$

and plot the accumulated results of all neurons across all 300 MVMs in Fig. 5.7. In (a) we plot the measured pump response, $a_{\text{meas}}^{(1)}$, against the theoretical linear value, $z_{\text{theory}}^{(1)}$, once again showing the nonlinear response due to the SA activation, and we maintain excellent agreement between experiment and theory. Here we keep the value of $E_{\text{sat}} = 0.19E_{\text{max}}$ consistent, however we observed the need to refit the optical depth, with $\alpha_0 = 9.4$ giving a better fit to theory. Note the errors around zero are suppressed, with no 'sign-flip' errors as we saw in the linear MVM-2a, as the amplitude is reduced to zero for the range of input values that produced these errors.

In Fig. 5.7(b) we plot the measured probe response, $\delta_{\text{meas}}^{(1)}$. The theory is shown by the red line. Once again, we found that the best fit between theory and experiment gave a different optical depth of $\alpha_0 = 7.3$, slightly off from the previously observed values. The variations in calculated optical depth could be due to the higher degree of noise and associated uncertainty in the numerical fitting. However it is also possible the optical depth of the cell drifted over the duration of data acquisition, most likely due to temperature variation. The saturation threshold, which is dependent only on the relative input power, is kept fixed throughout, and the optical depth was observed to be consistent over the length of time needed for optical training.

For comparison, we also plot the exact gradient of the SA function, as given by Eq. (5.15) (orange line), scaled arbitrarily to match the measured response. This

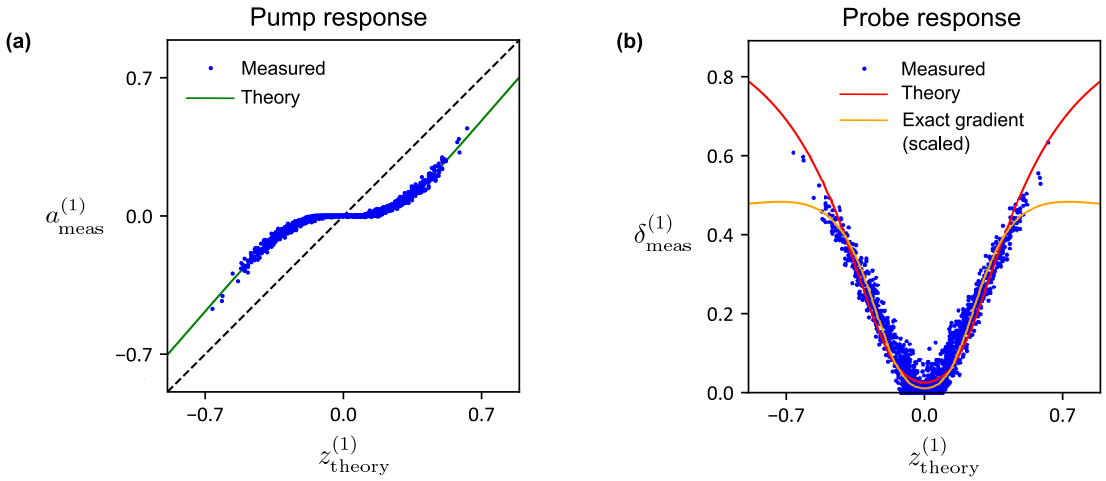


Figure 5.7: Measured pump and probe response when performing random real-valued MVM. (a) First-layer activations $a^{(1)}_{\text{meas}}$ measured after the vapor cell, plotted against the theoretically expected linear MVM-1 output $z^{(1)}_{\text{theory}}$ before the cell. The green line is a best fit curve of the theoretical SA nonlinear function. (b) The amplitude of a weak constant probe passed backwards through the vapor cell as a function of the pump $z^{(1)}_{\text{theory}}$, with constant input probe. Measurements for both forward and backward beams are taken simultaneously.

scaling factor can be compensated by changing the learning rate during the optical training. We see within a given range, of approximately $\pm 3E_{\text{sat}}$, the approximation between the probe response and the exact gradient is excellent. Matching this range to the actual values taken by the neurons during optical training is key, and why having sufficient power to over-saturate the cell is so critical.

5.2 Optical training of an ONN

5.2.1 Methods

All the necessary operations can now be performed to optically train a neural network, including forward and backward propagation through a two-layer network, with a hidden-layer activation function. Our network architecture is $N = 3, M = 5$ and $K = 2$, and we perform training on a nonlinear classification task. The inputs to the network are the coordinates (x_1, x_2) of each data point in a 2D plane. Each point belongs to one of two classes, Class I (blue circles) or Class II (orange triangles), and the output of the network is a prediction as to which class each data point

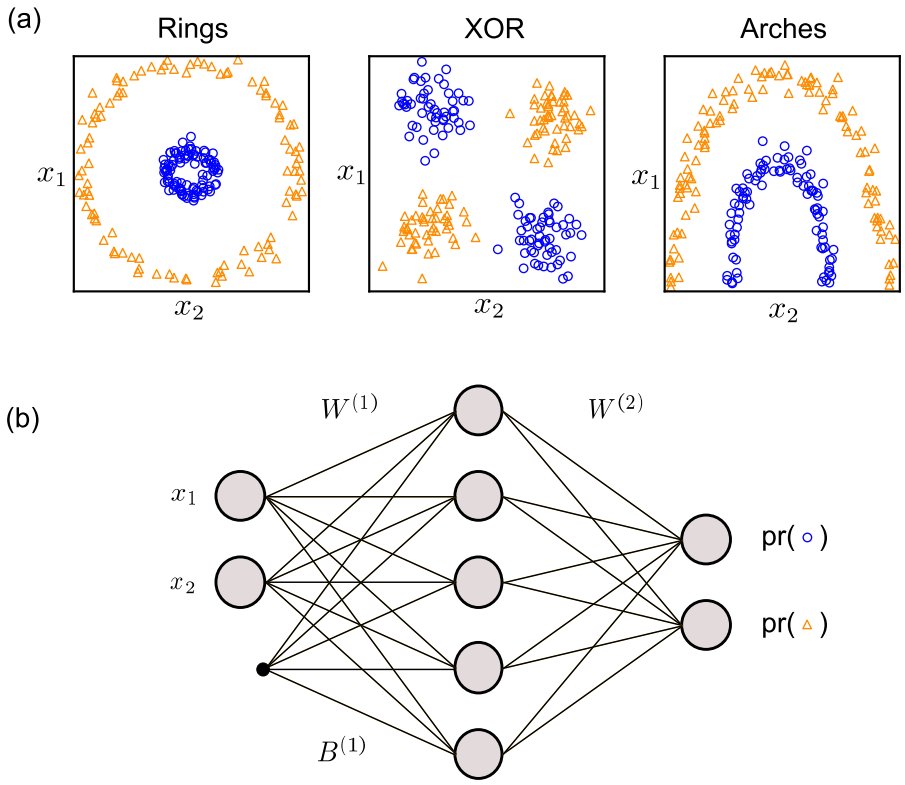


Figure 5.8: Three example datasets classified with optical training of the ONN. (a) Each dataset consists of two classes of objects with coordinates (x_1, x_2) in a 2D plane, separated with nonlinear geometries. (b) The neural network architecture of our ONN.

belongs. The data points are clustered with some nonlinear geometry, and the task for the network is to identify the boundary between the two classes.

We perform our optical training on three such geometries, which we refer to as 'rings', 'XOR' and 'arches' datasets. Each dataset consists of 600 points, balanced between the two classes, and they are shown in Fig. 5.8. Each dataset is split into training and validation sets of size 400 and 200. Our test set is different, and consists of a 20×20 uniform grid of input coordinates. By making a prediction of the class for all these grid coordinates, we can visualise how the network is drawing the boundary between the two classes.

Until now, we have simplified the discussion of feed-forward neural networks by omitting the use of 'biases' in each linear layer. These are tunable parameters equivalent to the weights, added to the results of each layer MVM. Therefore

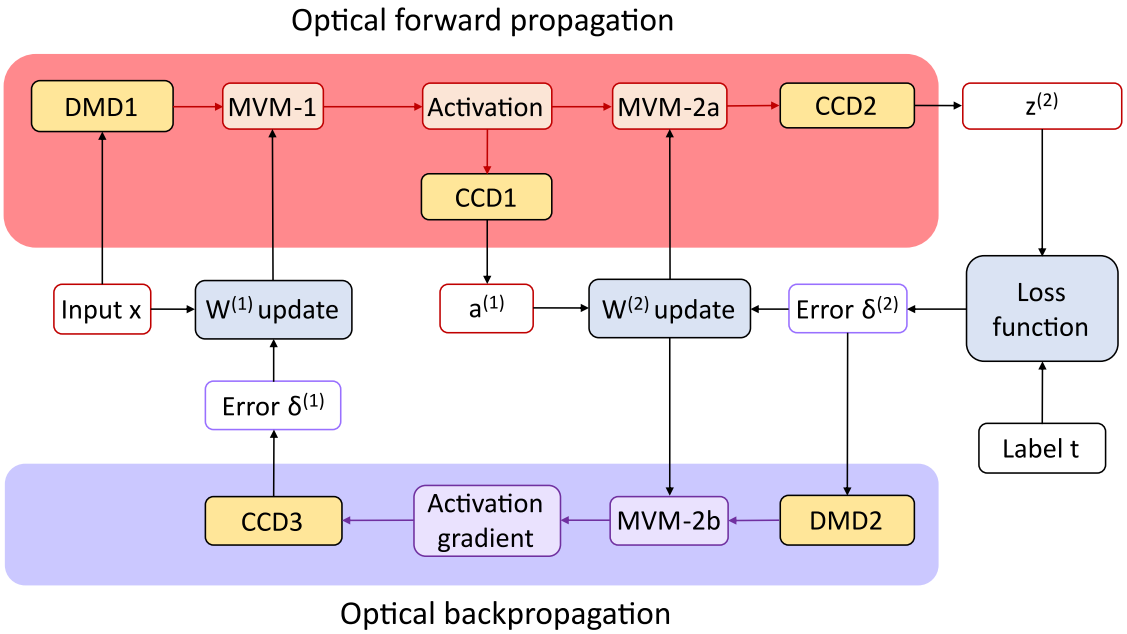


Figure 5.9: Flow of optical training scheme, including the nonlinear activation at the hidden layer. Duplicate of Fig. 4.6, with minor alteration to include activation and gradient.

each linear layer of the network becomes

$$z_j^{(l)} = \left(\sum_i W_{ji}^{(l)} a_i^{(l-1)} \right) + B_j^{(l)}, \tag{5.30}$$

where $B^{(l)}$ is the tunable bias vector at layer l.

The biases are important for the network to fully utilise the nonlinear activations, and so we introduce a bias to the first layer of our ONN (the output layer is sufficient with weight matrix only.) This is why we use $N = 3$; two input neurons represent (x_1, x_2), and the third remains constant. The first two columns of SLM1 then represent $W_{ji}^{(1)}$, whilst the third column represents $B_j^{(1)}$. Summation with the cylindrical lens then implements exactly Eq. (5.30).

The biases are tuned by backpropagation exactly as with the weights, calculating the gradient that minimises the loss function. This gradient is in fact

$$\frac{\partial \mathcal{L}}{\partial B_j^{(1)}} = \delta_j^{(1)}, \tag{5.31}$$

i.e. exactly the error vector being calculated for the weight update. So we can introduce this bias with no additional calculations needed in the training scheme.

Dataset	Input neurons	Hidden neurons	Output neurons	Learning rate	Epochs	Batches per epoch	Batch size
Rings	2	5	2	0.01	16	20	20
XOR				0.005	30		
Arches				0.01	25		

Table 5.1: Summary of network architecture and hyperparameters used in training the ONN for each dataset.

The optical training proceeds as outlined in Sec. 4.2.2, and we summarise again in Fig. 5.9 (a duplicate of Fig. 4.6 included for convenience, but now including the nonlinear activation and gradient steps). Each iteration consists of measuring $a^{(1)}, z^{(2)}$ and $\delta^{(1)}$ optically. Note in particular that $a^{(1)}$ is measured directly in the forwards pass, without need for calculating $z^{(1)}$. Similarly $\delta^{(1)}$ is measured directly in the backward pass, without need for calculating $\rho^{(2)}$ (albeit calculated with multiple measurements to remove background offsets and calculate the sign).

We perform the training with 20 mini-batches of size 20 per epoch, and we tune the learning rate and number of epochs to maximise the classification accuracy for each of the three datasets. The class labels t are one-hot encoded vectors $(1, 0)$ and $(0, 1)$, and we use softmax and CCE as the output activation and loss function.

5.2.2 Results

Learning curves

Figure 5.10 shows the optical training performance on the three datasets. In each case, we perform five repeated training runs, and plot the loss and accuracy for the validation data after each epoch of training (shown as mean value shaded up to one standard deviation). To visualise how the network is learning the boundary between the two classes, we run the test dataset after each epoch. The resulting 'decision boundary' plots after 1, 3, 6 and 10 epochs are also shown. We see that the ONN quickly learns the nonlinear boundary and gradually improves the accuracy to 100%. This is evidence the optical nonlinearity in the system is contributing to the network performance, and that the gradient approximation in the optical backpropagation is good enough to train the network.

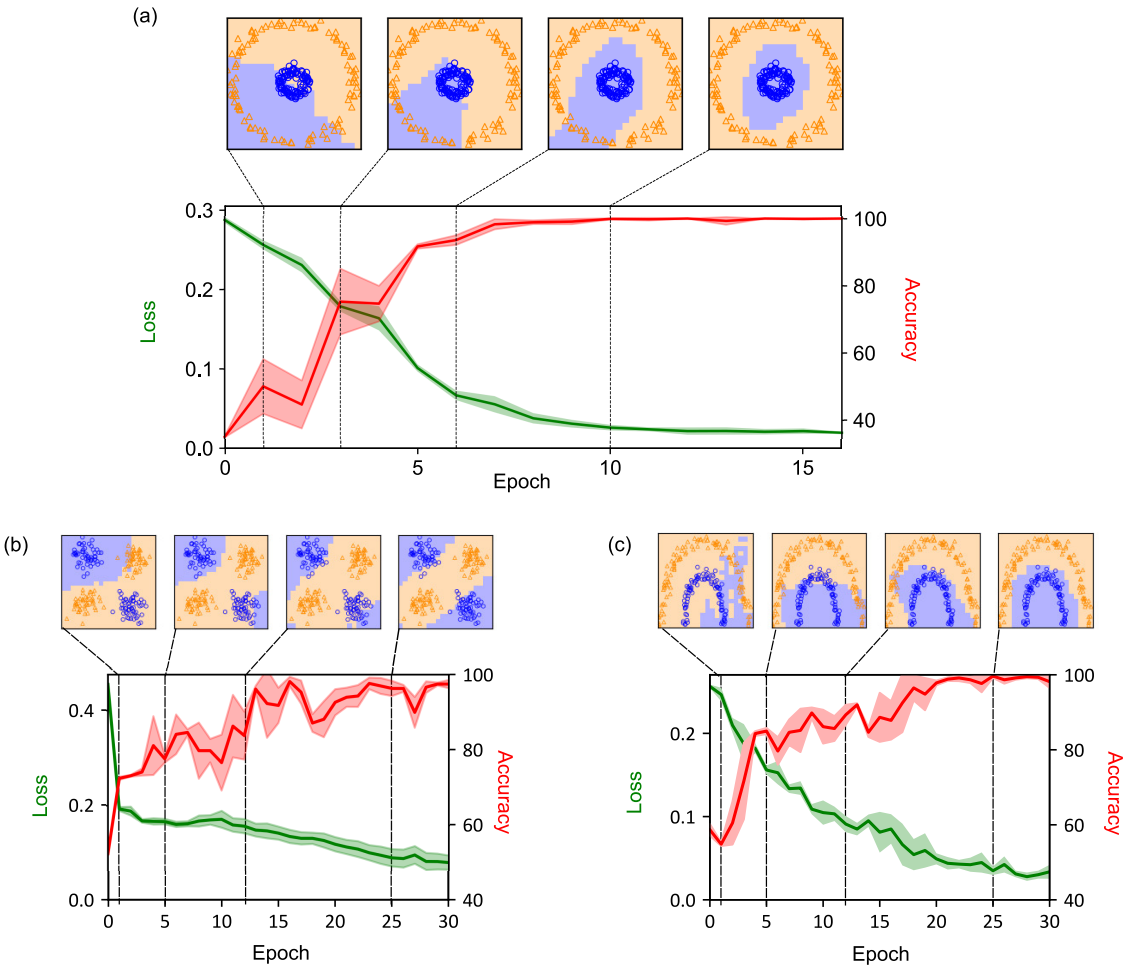


Figure 5.10: Learning curves and example decision boundary plots of the ONN for classification of each dataset. (a) Mean and standard deviation of the validation loss and accuracy averaged over 5 repeated training runs of the Rings dataset. Shown above are decision boundary plots of the ONN inference output for the test set, after different epochs. Equivalent plots for (b) XOR dataset and (c) Arches dataset.

The 'XOR' and 'Arches' datasets are significantly harder boundaries to learn, and the network requires a greater number of epochs to converge. Even then, the accuracy does not converge smoothly. This is predominantly a result of the small validation set, as small fluctuations in the decision boundary can cause a large change in the number of correctly predicted inputs. Regardless, both datasets still converge to nearly 100% accuracy.

It was also observed the weight initialisation was important for successful training. For some weight initialisations, the network training would flatline, and was unable to further reduce the loss function, even with tuning of the learning rate. This

was observed even with offline training, and is a consequence of the small network architecture. With only 5 neurons in the hidden layer, the network is not expressive enough to escape some local minima of the loss function. The results shown in Fig. 5.10 are averaged over five examples of successful training runs.

ONN precision

Since the network successfully converges, the gradients (and by extension the activation and error vectors $a^{(1)}$, $\delta^{(2)}$ and $\delta^{(1)}$) must have been calculated with sufficient accuracy. Here we analyse more quantitatively the discrepancy between the measured and expected values for the XOR dataset - empirically the hardest dataset to train.

We perform again the optical training, and this time digitally calculate the theoretical value of every vector throughout the training procedure. In Fig. 5.11 we plot the scatter of measured against theory values for all three vectors measured optically, for the first 5 epochs of training (after which the distributions were seen to stabilise). In the first two columns we plot the results for $a^{(1)}$ and $z^{(2)}$. In general the SNR is excellent, although for $a^{(1)}$ some 'sign-flip' errors are present very close to zero, and an offset at values around -0.2 is evident. This indicates a slight deviation from the numerically fit SA nonlinear curve $g(\cdot)$, potentially due to a change in optical depth from temperature fluctuations.

In the final column we plot the results for $\delta^{(1)}$, the backpropagated error. Recall this vector is a function of two variables: the forward 'pump' from MVM-1, $z^{(1)}$, and the backward 'probe' from MVM-2b, $\rho^{(2)}$. The SNR is clearly smaller than the forward vector results, as we might expect given the complex pump-probe interaction. The peculiar shape of the scatter plot is a result of how the sign is measured, as outlined in Sec. 5.1.3. The benefit of this scheme is that almost no error vector elements are measured with the incorrect sign, which is extremely important for stable network training [166]. Finally, we also see the range of values taken by the backpropagated error reduce within the first few epochs, as one should expect as the weights converge and loss function reduces.

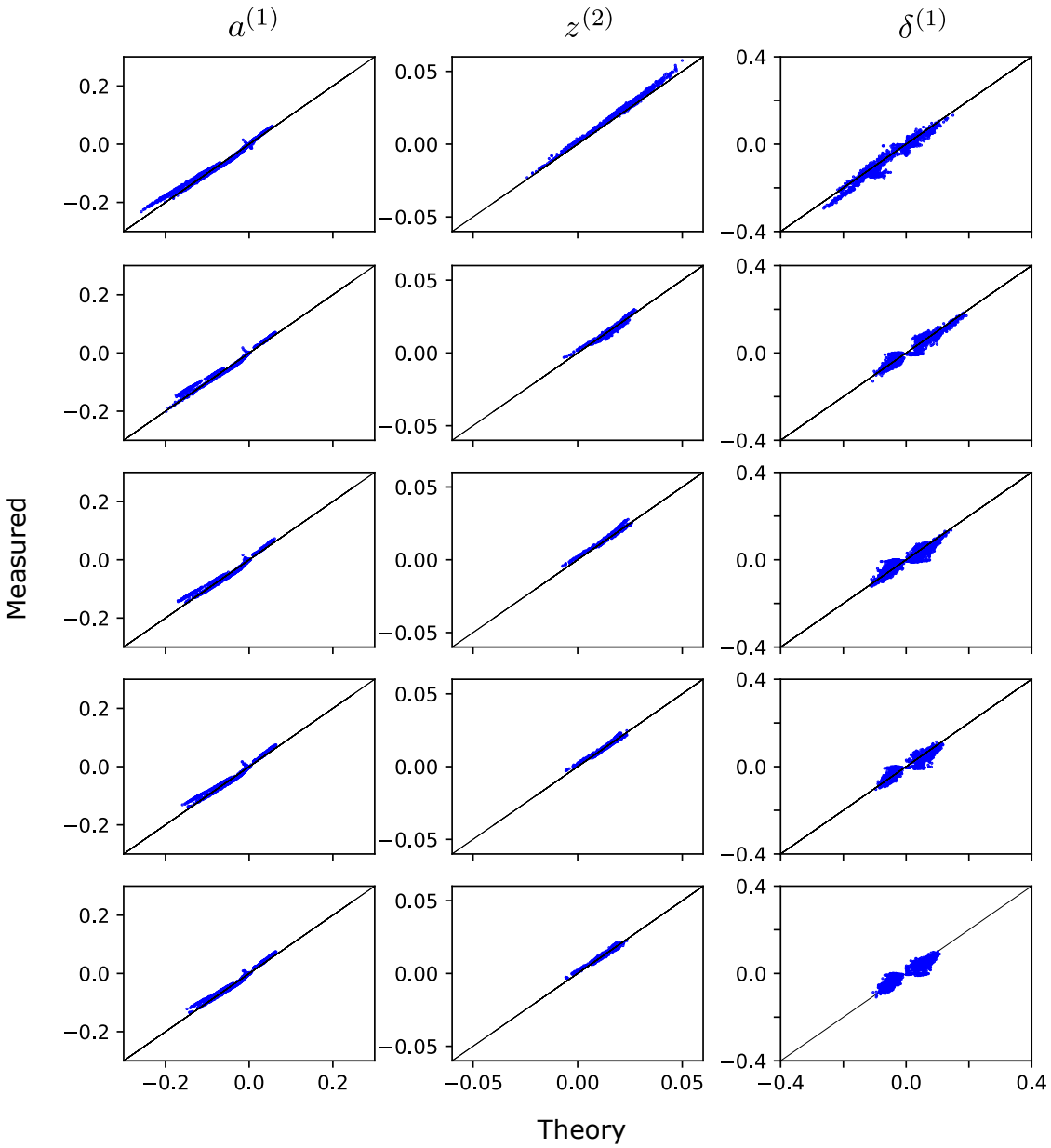


Figure 5.11: Evolution of measured against theory scatter plots over the first five epochs of optical training. Each plot includes all vector elements from every mini-batch in each epoch. Left: hidden-layer activation vector; middle: output vector; right: backpropagated error vector.

In Fig. 5.12 we plot the same results for the backpropagated error in the first epoch only, separated by mini-batch, and colour the datapoints by neuron, i.e. by vector element $\delta_j^{(1)}$. We see that initially all the errors are predominately negative, and certain neurons have much larger errors than others. As the weights are updated, the error for each neuron starts to spread, and centre around zero.

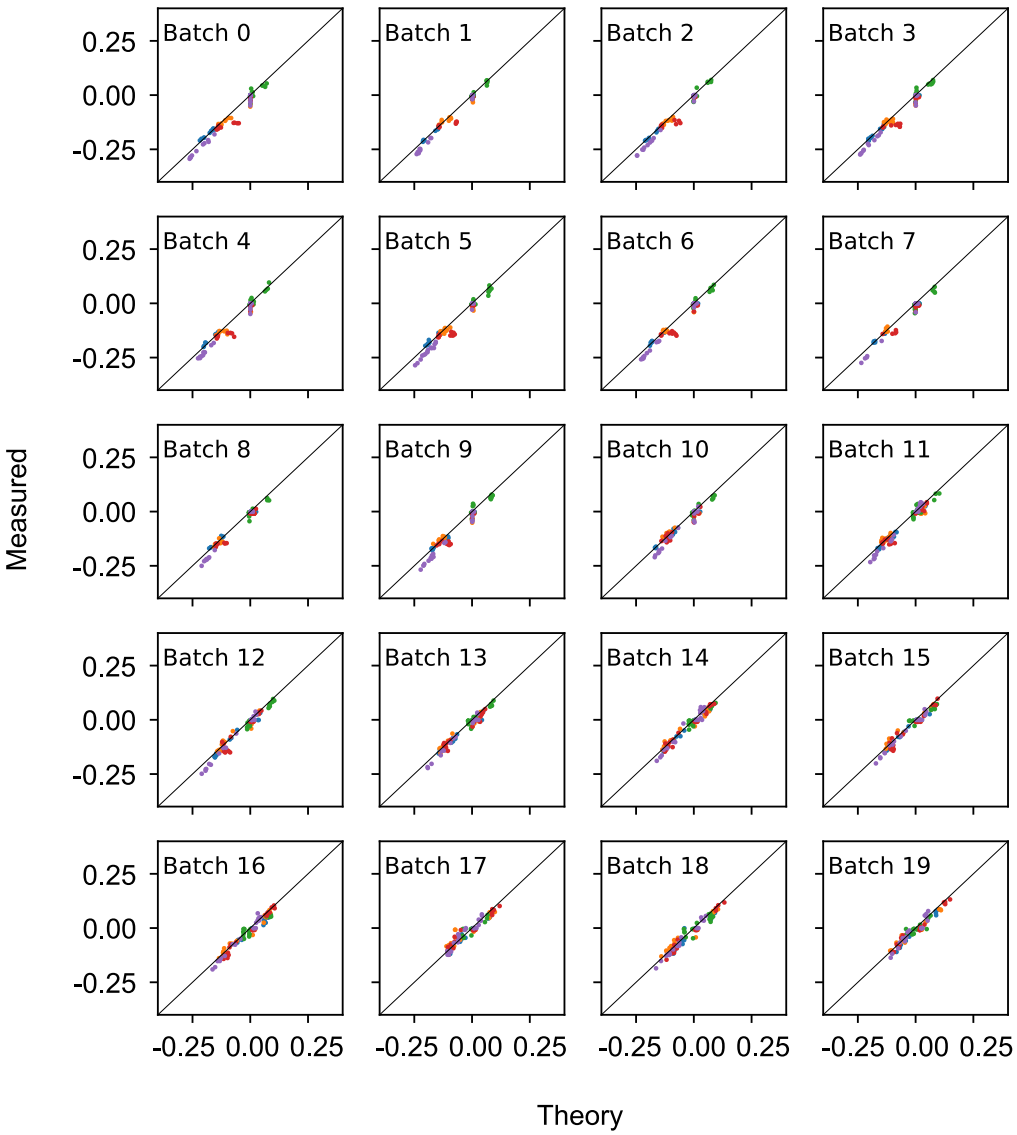


Figure 5.12: Evolution of backpropagated error distribution during the first epoch of optical training. A scatter of measured against theory value for each training mini-batch is shown. Data points for each neuron are distinguished by colour.

This is again strong evidence that the correct error vectors are being optically calculated, and so the proper parameter gradients and updates are being applied to the network weights and biases.

Optical vs. offline training

Finally, we compared the performance of our optical training scheme against offline training. Whilst offline training has the benefit of being able to use the full bit-precision of a digital computer to calculate the gradients, we have already observed

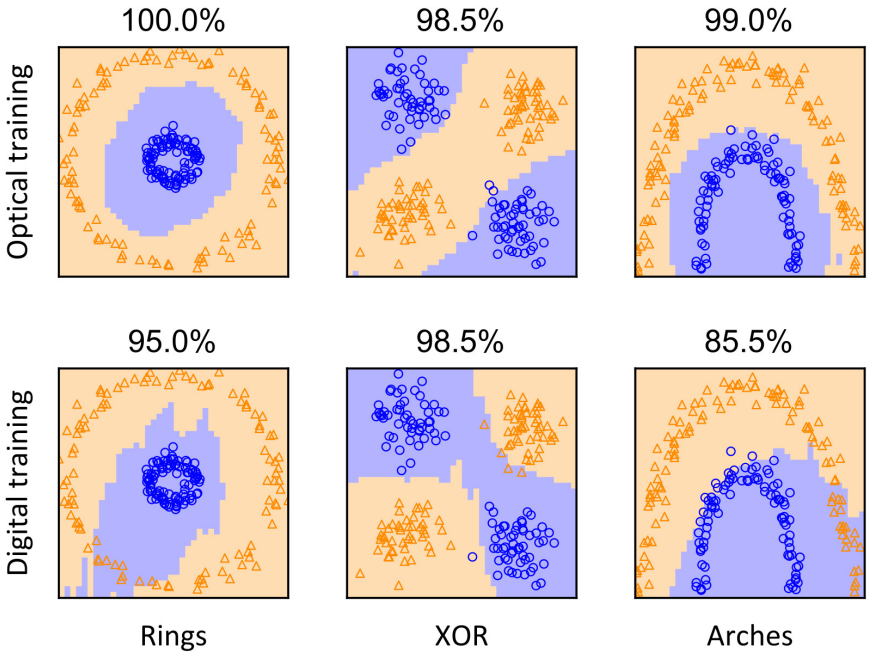


Figure 5.13: Decision boundary plots of the ONN inference output on the test set for the three different datasets, after the ONN has been trained optically (top) and offline (bottom).

in numerous cases that hardware-in-the-loop training schemes can be advantageous over digital training in overcoming the 'reality-gap'.

We digitally model our system with a neural network of the equivalent architecture, including identical learning rate, number of epochs and all other hyperparameters. In this model, the hidden-layer nonlinearity $g(\cdot)$ and the associated gradient are given by the best fit curve and theoretical probe response of Fig. 5.7, and the weights and biases are trained using the same training set.

The offline-trained parameters are then used for inference with the ONN, performed on the test set of uniformly spaced grid coordinates. The top and bottom rows in Fig. 5.13 plot the decision boundary, after the system has been trained optically and digitally respectively. For all three datasets the optically trained network achieves almost perfect accuracy, as presented above. However the boundary learned by the digitally-trained network clearly does not match the ground truth data. In all cases the resulting classification accuracy is reduced, and significantly so for the arches dataset, achieving only 85%. These results once again

reinforce the advantage of hardware-in-the-loop training schemes over offline training.

5.3 Conclusion

In this final experiment, we have demonstrated for the first time end-to-end optical backpropagation through a nonlinear activation function. The propagation of an optical beam represents a linear transformation, such that optics is well-suited to implementing the linear interconnection of neurons in a network, but including a nonlinearity is essential for neural networks to learn complex mappings. This presents a challenge for designing all-optical neural networks, since nonlinear optical phenomena are less prevalent and more complex to implement.

Our optical training scheme overcomes both this and the even more challenging problem for nonlinear layers in an ONN: calculating the derivative of the activation during backpropagation. By successfully training various classification problems with our end-to-end optical training scheme, we have shown that the saturable absorption mechanism is suitable as the nonlinear activation function in an ONN, simultaneously providing the correct transformation in both forward and backward directions.

However it is important to acknowledge the limited scope of our experiment, having implemented only one optical nonlinear activation. It is unclear from this experiment alone how well such a mechanism scales to multiple layers in a deep network. Of primary concern is the accumulation of an offset in the backpropagation signal. In this experiment we took multiple measurements in order to remove various background fields, which were subtracted digitally in order to obtain the desired signal. However such background terms are likely to accumulate when propagating through multiple vapor cells, eventually limiting the dynamic range of the signal. This may represent a bottleneck when scaling to larger and deeper ONNs.

Regardless, saturable absorption is likely not the most appropriate nonlinear mechanism for use in a deep network, due to significant loss of optical power at each layer. Using saturable gain would negate this issue, and warrants further investigation as the nonlinearity in our optical training scheme. Using erbium-doped fiber amplifiers (EDFA) to connect layers of a network is an obvious way to

implement this in practice. Further analysis on offset accumulation in this setting is required, as well as considerations of the achievable energy efficiency, with optical loss from fibre coupling and larger energy consumption from the amplification process possible limitations for scalability.

6 Conclusion

Contents

6.1 Summary

This thesis has detailed the first demonstration of a multi-layer optical neural network, with end-to-end optical training. Our ONN is a physical implementation of an artificial neural network, the most important class of machine learning algorithm that underpins an incredible range of modern technology.

The foundation of the ONN is an optical multiplier performing optical matrix-vector multiplication. This was built using spatial light modulators, such as DMDs and LC-SLMs, and we explored the encoding methods, coherent detection schemes, and calibration procedures needed to perform precise MVM with both real-valued and complex-valued parameters. We created a multiplier that supports a matrix size up to 200×50, one of the largest demonstrated that is real-valued, reconfigurable and performs fan-in and fan-out.

We used our multiplier as a simple linear classifier, as well as a hybrid opto-digital network, and a complex-valued network. Using these ONNs we demonstrated hybrid training, a hardware-in-the-loop training scheme that helps mitigate the reality-gap problem often seen when training analog or physical hardware with a digital simulator. Throughout this work we used supervised learning, whereby network parameters are iteratively updated to find their optimal values using the backpropagation algorithm. Using our hybrid training scheme, where the forward pass through the network is performed with the physical ONN, we were able to achieve good classification accuracy on the MNIST dataset.

Multiple MVMs can be cascaded to construct a multi-layer ONN, and we created a two-layer network in experiment using additional LC-SLMs, narrow slits and cylindrical lens sets. Additionally, optical backpropagation was implemented by introducing a second beam to pass backwards through the experiment. We successfully trained the two-layer ONN with our optical training scheme, where both the forward inference-like step and error backpropagation steps of the training algorithm are performed with the physical ONN.

Finally, we implemented an optical nonlinear activation function at the hidden-layer of our ONN, using the phenomenon of saturable absorption in a rubidium vapor cell. Crucially, this nonlinearity supported our optical backpropagation scheme, allowing us to optically calculate the gradients of the nonlinear function, using a pump-probe scheme similar to that used in doppler-free saturation spectroscopy. Using our nonlinear two-layer ONN and optical training, we performed classification on a range of nonlinear decision boundary tasks, which significantly outperformed offline training.

6.2 Discussion and outlook

We have shown through numerous examples the benefits of online training, with our hybrid and fully-optical schemes helping overcome the reality-gap between physical system and digital model, and ultimately improve the network accuracy of our ONNs. We have used SLM-based MVM as the linear neuron interconnection, and saturable

absorption in a vapor cell as the nonlinear activation function, to demonstrate our optical backpropagation scheme. However the optical training scheme is not limited to this implementation, and can in fact be used with a variety of methods.

Alternative methods to perform optical MVM, such as the photonic crossbar array, are also bidirectional, in that the optical field propagating backwards gets multiplied by the transpose of the weight matrix. This would therefore be a natural implementation of our scheme with integrated photonics. Beyond MVM, diffraction naturally works in both directions, so diffractive neural networks also satisfy the requirements. Optical convolution, achieved by performing Fourier transforms with a simple optical lens, and mean pooling, achieved through an optical low pass filter, also work in both directions, and so a convolutional network can be optically trained as well.

The critical requirement of the optical nonlinearity is the ability to acquire gradients during backpropagation. Our pump-probe method is compatible with multiple types of optical nonlinearities: saturable absorption (as demonstrated in our experiment), saturable gain and intensity-dependent phase modulation [167]. Using saturable gain as the nonlinearity would offer the added advantage of loss compensation in a deep network, and using intensity-dependent phase modulation nonlinearity, such as self-lensing, would naturally support complex-valued optical neural networks.

The value and potential of hardware-in-the-loop training schemes for future analog devices is clear, but they have so far only been studied at a small scale. The review by Buckley et al. [124] raises this exact question: do such schemes still provide benefit as networks become large enough to address real-world problems? Many alternatives to hardware-in-the-loop training are being explored, including digital noise-aware training, and training networks with lower precision arithmetic. As the precision of analog hardware increases, and the required precision of network training falls, in-the-loop schemes may not be required.

Additionally, in-the-loop training necessitates that inference of an analog neural network is performed on the same hardware as it was trained. In many real

world applications this is not the case. Consider the example of fully autonomous cars, an extremely challenging problem that requires an enormous and complex neural network to be trained with a large computing cluster. After the network is trained, the model parameters are uploaded to the computer onboard each vehicle, which uses the trained model to perform inference on sensor data and drive autonomously. It would clearly be unfeasible that an analog processor on board each vehicle is individually trained from scratch. Far more likely is an approach closer to 'fine-tune' training, where a pre-trained model is individually optimised for use on the inference hardware.

However, the potential benefit in speed and energy efficiency from using analog processors, and optical processors in particular, is undeniable. The growth in global demand for computer processing power, as a direct result of the advancements in machine learning and neural networks, is unprecedented. Sustainable alternatives for neural network processing are required, not just for inference, but also for training. Further development of online training for ONNs (and other analog hardware) is not just about addressing issues such as the reality-gap problem, but about creating fundamentally more scalable and more sustainable approaches to computing.

There are many potential approaches to implement a wide variety of neural networks in optics, and it is still not clear which approach will prove the most successful. Equally, it is unlikely optical computing will converge to a single technology, with different forms better suited to different environments and types of problem. Whilst our experiments with free-space optics clearly remain proof-of-principle demonstrations, they help identify some of the major hurdles that remain in order to scale the technology to real-world workloads. These include multiplier precision, speed of weight-update, and the achievable number of neurons and layers.

We demonstrated in our experiments a multi-layer ONN with sufficient signal-to-noise ratio to train a simple classification task. However modern ANNs trained on digital platforms primarily use floating-point arithmetic, which allows a broad dynamic range of values, and crucially gives high precision around zero. Analog processors are equivalent to integer arithmetic with relatively low bit-precision, and

therefore appear to be less suitable for training large and deep neural networks. However, it is already common to quantize pre-trained ANNs to use integer arithmetic during the inference stage, in order to reduce the needed compute power and increase energy efficiency [168]. A natural continuation is to use integer arithmetic for training ANNs as well, to which a great deal of research is currently dedicated [169, 170]. At the same time, the precision of photonic processors is continuously improving, with recent demonstrations achieving the equivalent of over 9-bit digital systems [171]. Some groups are also exploring how noisy analog systems can be exploited, inspired by stochastic neural networks [172]. In this case the multiplier imprecision is not a problem to be addressed, but a benefit to be utilised.

Whereas inference does not require the weights to be updated or tuned, training by its very definition relies on regular and rapid parameter updates. In this case, the speed of electro-optical modulation technologies is critical to scaling ONNs for real-world use. Current SLM technology for free-space ONNs, and thermo-optical modulators in integrated photonics, are limited to slow kHz speeds, but much faster modulation techniques are being developed, including novel SLM designs [173], carrier injection and depletion technology [174], phase-change materials [111] and electromechanics [175]. Scalable ONNs with modulator speeds above 100GHz are already within reach.

Further increasing the number of neurons and layers in a fully optical ONN remains challenging. Whilst individual optical multipliers have been demonstrated with 1000s of neurons, cascading such multipliers to form deep networks poses difficulties. Issues arise from noise accumulation, reduced multiplier precision and significant optical loss. Indeed our experiments demonstrated a very high degree of optical loss, and were limited to just two layers. The use of saturable gain as the optical activation function could be promising to overcome the latter, as it simultaneously provides the nonlinear function required of the activation between layers, but also serves to amplify the signal throughout the ONN.

The continuous improvement of silicon-based digital processors over the past half century has been remarkable, however they too face challenges in continuing to

improve speed and energy efficiency. Combined with the ever-increasing demand for larger and more complex AI models, this presents a unique opportunity for optical computing. The most important metric for all types of analog hardware is scalability, and for any implementation of optical computing to succeed it is crucial we start to move beyond proof-of-principle demonstrations. Utilising optical fan-out in free-space optics may offer a scaling advantage in number of arithmetic operations performed per digital-to-optical conversion, but we may still require technology or material breakthroughs to reach sufficient scale to realise the benefit. The need for a new type of SLM with greater pixel resolution and refresh speed is an obvious example.

Other implementations of optical computing based on integrated photonics may offer energy efficiency benefits, as they avoid issues of energy loss from optical fan-in, but are again constrained by the limits of fabrication technology today. Developing new photonic components with significantly smaller footprint will be essential, in order to increase the density of on-chip components.

Finally, future improvements in AI and neural networks will not derive from advances in hardware alone. The development of new neural network architectures and algorithms that are well suited to the analog hardware on which they are run will be essential. For some applications these may be novel neuromorphic-type algorithms that require new training algorithms. However given the incredible success of the backpropagation algorithm in training so many modern AI architectures, it seems pertinent to explore how it can be implemented with analog hardware, and for the first time our experiments demonstrate the feasibility of end-to-end optical backpropagation in ONNs.

6.3 Future work

Some avenues explored during the course of these experiments, but ultimately not pursued, could significantly improve our optical training scheme. First, providing an external phase-locked reference beam would allow real-valued detection with a single-shot measurement at all layers simultaneously. This would in principle

simplify the multiple different measurement schemes used in our optical training experiment, but significantly increase the experiment complexity in practice.

Initial testing of such a single-shot measurement with external reference did demonstrate the possibility for a single MVM. A separate reference beam was generated, and overlapped with the signal at the CCD plane. The signal and reference interference was measured by a photodetector, and a locking mechanism was created using an Arduino microcontroller. This implemented a feedback loop between the measured interference and a piezo-controlled actuator to vary the path length (and therefore relative phase) of the reference beam. Measuring the intensity of the interference pattern between output vector signal and reference for a random MVM over time reveals the stability of the phase locking. Example results are shown in Fig. 6.1, showing that extreme intensity fluctuations without the locking mechanism are removed when the locking mechanism is turned on. However maintaining a similar level of phase-lock stability over the full duration of hybrid or optical training was extremely challenging, and therefore not implemented in the experiments. Successfully implementing single-shot, real-valued detection for all layers of a coherent ONN will be important when scaling to larger and deeper networks, and will likely require a more sophisticated form of phase locking than investigated here.

Second, all the tasks demonstrated in the experiments in this thesis were classification problems, and were therefore labelled with onehot encoding vectors, which are by definition binary. Regression problems, and other types of task where the label or output target takes continuous values, are more challenging for an ONN as the network accuracy depends on the multiplier precision to a greater extent. We chose not to explore regression tasks in these experiments as they require only a single output neuron, and were therefore not able to demonstrate true MVM in the second layer. However it will be important to verify that ONNs, and our optical training scheme, can be applied to a variety of tasks beyond classification.

Many additional avenues exist to further the development of ONNs more broadly, and explore the applicability and possibilities of our optical backpropagation scheme.

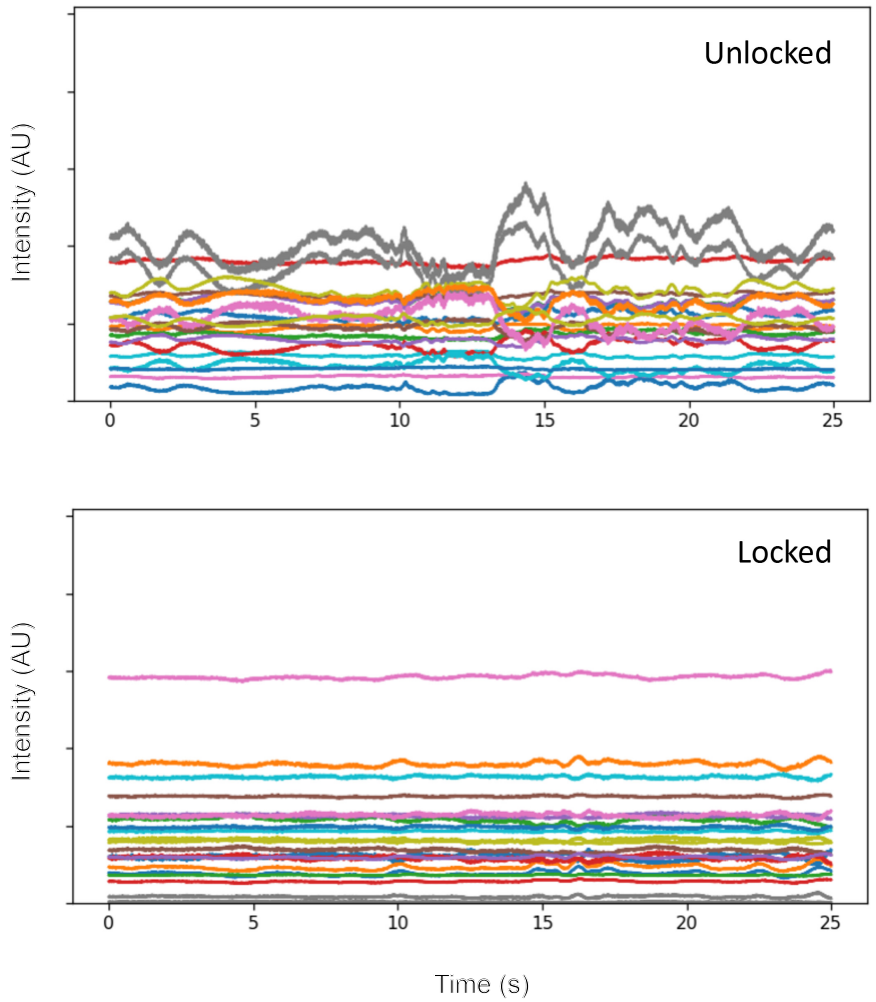


Figure 6.1: Initial results showing phase stability between MVM signal and an external reference beam, with and without phase locking. Each vector element is plotted individually with arbitrary colour.

First, the use of saturable gain as the nonlinearity is more promising for use in a deeper optical network, where the ability to amplify the optical signal between layers and mitigate against optical loss will be crucial. The use of erbium-doped fiber amplifiers (EDFA) to connect layers of a network is a promising route to implement saturable gain in practice.

Second, the natural ability of light to encode complex-valued weights, utilising the ability to encode information in both the amplitude and phase of coherent beam, is a distinct advantage over conventional electronics. We briefly demonstrated this ability in ONN-3 of our hybrid training experiments, and it deserves further

investigation. Implementing our optical backpropagation scheme with complex-valued weights would require implementing complex conjugation of the weight matrix in the reverse direction - a challenge without obvious solution to implement in practice - but would allow for far more efficient training of complex-valued networks. Beyond training, complex-valued ONNs for inference applications that require only intensity measurement, rather than full coherent detection schemes, may be easier to implement and scale.

Finally, a growing field of research is focused on using optical processors in applications where the input data is already in the optical domain, such as imaging and object detection. Eliminating the need for digital to optical conversion brings a variety of benefits, however in practice most applications would require the processing of incoherent light. The use of incoherent sources in ONNs, where the ability to encode real-valued or complex-valued weights is lost, remains an open challenge, especially for training such ONNs.

Appendices

A

Common loss functions and their derivatives

Contents

Here we derive the equations of the first gradient required for backpropagation for two common loss functions: Mean-squared error (MSE) and categorical cross-entropy (CCE). Further details can be found in for example [176].

Recall we describe, for a network with L layers, the final linear result as the vector $z_j^{(L)}$. We may apply an activation function to this vector to obtain the network output y_j.

For backpropagation, we require the error vector

$$\delta_j^{(L)} = \frac{\partial \mathcal{L}}{\partial z_j^{(L)}} = \frac{\partial \mathcal{L}}{\partial y_k} \frac{\partial y_k}{\partial z_j^{(L)}}. \tag{A.1}$$

A.1 Mean squared error

MSE is a typical loss function used when solving regression problems. The definition of MSE loss between network output y and label t is

$$\mathcal{L} = \frac{1}{2}\sum_i (y_i - t_i)^2. \tag{A.2}$$

It is also common for no activation function to be applied to the linear output of the final layer of a network when using MSE. Therefore the network output for a network with L layers is simply

$$y_i \equiv z_i^{(L)} \tag{A.3}$$

and the error vector is (trivially)

$$\delta_i^{(L)} = z_i^{(L)} - t_i. \tag{A.4}$$

A.2 Categorical cross-entropy

CCE is commonly used in conjunction with the softmax activation function, when solving classification problems. The softmax activation function gives

$$y_i = \frac{e^{z_i}}{\sum_d e^{z_d}}, \tag{A.5}$$

and the CCE loss function is defined as

$$\mathcal{L} = -\sum_i t_i \log\left(y_i\right), \tag{A.6}$$

where t_i is the binary one-hot label.

We require

$$\delta_i^{(L)} = \frac{\partial \mathcal{L}}{\partial z_i} \tag{A.7}$$

$$= -\sum_k t_k \frac{1}{y_k}\frac{dy_k}{dz_i}, \tag{A.8}$$

using the standard chain rule of calculus.

The terms of the sum $\frac{dy_k}{dz_i}$ are calculated with the quotient rule in two cases. If $i = k$:

$$\frac{dy_k}{dz_i} = \frac{\sum_d e^{z_d} \cdot e^{z_k} - e^{z_k} \cdot e^{z_k}}{(\sum_d e^{z_d})^2} \tag{A.9}$$

$$= \frac{e^{z_k}}{\sum_d e^{z_d}} \cdot \left(\frac{\sum e^{z_d} - e^{z_k}}{\sum_d e^{z_d}} \right) \tag{A.10}$$

$$= y_k \cdot (1 - y_k). \tag{A.11}$$

If $i \neq k$:

$$\frac{dy_k}{dz_i} = \frac{\sum_d e^{z_d} \cdot 0 - e^{z_k} \cdot e^{z_i}}{(\sum_d e^{z_d})^2} \tag{A.12}$$

$$= -\frac{e^{z_k}}{\sum_d e^{z_d}} \cdot \frac{e^{z_i}}{\sum_d e^{z_d}} \tag{A.13}$$

$$= -y_k y_i. \tag{A.14}$$

Using these results, we therefore rewrite Eq.(A.7) as

$$\delta_i^{(L)} = \frac{\partial \mathcal{L}}{\partial z_i} \tag{A.15}$$

$$= \sum_k \begin{cases} t_k(y_k - 1) & i = k \\ t_k y_i & i \neq k. \end{cases} \tag{A.16}$$

Due to the one-hot encoding of t_k, by definition only one of the elements is equal to one, the rest are zero, so we can then further simplify to

$$\delta_i^{(L)} = y_i - t_i. \tag{A.17}$$

B

Generating complex-valued fields with LC-SLM phase grating

Contents

B.1 Overview

For simplicity we consider a one dimensional LC-SLM that is continuous in dimension x, with no pixelation. We want to produce the field

$$s(x) = a(x)e^{i\phi(x)}, \tag{B.1}$$

where a and ϕ are fixed functions of x. However an LC-SLM can only transform the field as

$$E(x) = e^{i\psi(x)}. \tag{B.2}$$

Our aim is to find a form of $\psi(x)$ that will give $s(x)$ as the output.

B.2 Fourier Decomposition

We will first show that

$$e^{iA(x)\theta(x)} = \sum_{q=-\infty}^{\infty} e^{-i(q-A(x))\pi}\text{sinc}[q - A(x)]e^{iq\theta(x)}. \tag{B.3}$$

To prove this we need two useful identities:

$$\text{sinc}[x] = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{ixt}dt, \tag{B.4}$$

$$\frac{1}{2\pi}\sum_{k=-\infty}^{\infty} e^{ikx} = \sum_{n=-\infty}^{\infty} \delta(x - 2\pi n) \tag{B.5}$$

Proof.

$$\begin{aligned}
e^{iA(x)\theta(x)} &= \int_0^{2\pi} \delta(z + \theta(x))e^{-A(x)z}dz && \text{(B.6)}\\
&= \sum_{n=-\infty}^{\infty}\int_0^{2\pi} \delta(z + \theta(x) - 2\pi n)e^{-iA(x)z}dz && \text{(B.7)}\\
&= \int_0^{2\pi}\left(\sum_{n=-\infty}^{\infty} \delta(z + \theta(x) - 2\pi n)\right)e^{-iA(x)z}dz && \text{(B.8)}\\
&= \int_0^{2\pi}\left(\frac{1}{2\pi}\sum_{q=-\infty}^{\infty} e^{iq(z+\theta(x))}\right)e^{-iA(x)z}dz && \text{(B.9)}\\
&= \frac{1}{2\pi}\sum_{q=-\infty}^{\infty}\int_0^{2\pi} e^{iz(q-A(x))}dz\ e^{iq\theta(x)} && \text{(B.10)}\\
&= \sum_{q=-\infty}^{\infty} e^{-i(q-A(x))\pi}\left(\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{i(q-A(x))t}dt\right)e^{iq\theta(x)} && \text{(B.11)}\\
&= \sum_{q=-\infty}^{\infty} e^{-i(q-A(x))\pi}\text{sinc}[q - A(x)]e^{iq\theta(x)}. && \text{(B.12)}
\end{aligned}$$

□

B.3 Creating image

If we naively display on the SLM the grating pattern

$$\psi(a, \phi) = A(x)\text{mod}_{2\pi}\left[\phi(x) + u_0 x\right], \tag{B.13}$$

then using the above identity the field gets transformed as

$$\begin{aligned}
h(x) &= e^{iA(x)\text{mod}_{2\pi}[\phi(x)+u_0x]} && \text{(B.14)}\\
&= \sum_{q=-\infty}^{\infty} C_q(x)e^{-iqu_0x}, && \text{(B.15)}
\end{aligned}$$

where

$$C_q(x) = e^{-i(q-A(x))\pi}\ \mathrm{sinc}\left(q - A(x)\right) e^{iq\phi(x)}, \tag{B.16}$$

consisting of three terms: an 'annoying' A-dependent phase term, the amplitude term, and the target phase.

We take the Fourier transform, achieved by passing through a spherical lens, to obtain

$$\begin{aligned} H(u) &= \mathscr{F}[h(x)](u) && \text{(B.17)}\\ &= \mathscr{F}\left[\sum_{q=-\infty}^{\infty} C_q(x) e^{-iqu_0 x}\right](u) && \text{(B.18)}\\ &= \sum_{q=-\infty}^{\infty} \mathscr{F}\left[C_q(x)\right](u) * \delta\left(u - qu_0\right). && \text{(B.19)} \end{aligned}$$

where we have used the linearity of Fourier transform. Notice each term in the sum are shifted by different integer numbers of u_0. This is because the term $u_0 x$ is making a phase grating, diffracting the beam into infinitely many orders.

If we place a spatial filter in the Fourier plane, we can select just the first order $q = 1$. This means we retain only the field

$$H_1(u) = \mathscr{F}[C_1(x)](u). \tag{B.20}$$

By passing through another lens, we perform the inverse Fourier transform, and we are left with

$$\begin{aligned} h_1(x) &= C_1(x) && \text{(B.21)}\\ &= \mathrm{sinc}\left(1 - A(x)\right) e^{i[\phi(x) - (1-A(x))\pi]}. && \text{(B.22)} \end{aligned}$$

We wanted $s(x) = a(x)e^{i\phi(x)}$. Therefore we must make the substitutions

$$A(x) \to M(x) = 1 - \mathrm{sinc}^{-1}[A(x)], \tag{B.23}$$

and

$$\phi(x) \to F(x) = \phi(x) + \left(1 - M(x)\right)\pi. \tag{B.24}$$

B.4 Summary

We encode on the LC-SLM a phase grating

$$\psi(x) = u_0 x$$

, so that we get many diffraction orders in the Fourier plane. We then modulate the grating height and offset according to

$$\psi(x) = M(x) \mathrm{mod}_{2\pi} \left[F(x) + u_0 x \right], \tag{B.25}$$

where

$$M(x) = 1 - \mathrm{sinc}^{-1}[A(x)], \tag{B.26}$$

and

$$F(x) = \phi(x) + (1 - M(x))\, \pi. \tag{B.27}$$

This then correctly sets the desired amplitude and phase in the first diffraction order. We remove all other orders, so we precisely obtain the desired pattern.

We obtain the desired pattern if the bandwidth of the target pattern $s(x)$ is smaller than u_0, i.e. the scale of smallest detail in $s(x)$ is larger than the grating period.

C

Methods of Coherent Detection

Contents

C.1 Complex-valued measurement

Consider one element of the complex-valued MVM output z_j. We perform intensity measurement of the coherent sum of the complex-valued signal and reference regions, yielding

$$I_\theta = \left| E_s e^{i\phi_s} + E_R e^{i\phi_\theta} \right|^2 \tag{C.1}$$

where E_s and ϕ_s are the field amplitude and phase of the MVM result (signal region), E_R is the (arbitrary) field amplitude of the reference region, and ϕ_θ is the reference region phase, which can be set arbitrarily using the LC-SLM. We can expand (D.1) to

$$I_\theta = |E_s|^2 + |E_R|^2 + 2E_s E_R \cos(\phi_s - \phi_\theta). \tag{C.2}$$

We want to obtain the values

$$\mathrm{Re}(z_j) = E_s \cos(\phi_s), \tag{C.3}$$

and

$$\mathrm{Im}(z_j) = E_s \sin(\phi_s), \tag{C.4}$$

and do so by performing two pairs of intensity measurements. First we set $\phi_\theta = 0$ and $\phi_\theta = \pi$, and take the difference:

$$\begin{aligned} I_0 - I_\pi &= |E_s|^2 + |E_R|^2 + 2E_s E_R \cos(\phi_s) && \text{(C.5)} \\ &\quad - \left(|E_s|^2 + |E_R|^2 + 2E_s E_R \cos(\phi_s - \pi)\right) && \text{(C.6)} \\ &= 4E_s E_R \cos(\phi_s) && \text{(C.7)} \\ &= 4E_R \mathrm{Re}(z_j). && \text{(C.8)} \end{aligned}$$

Next we set $\phi_\theta = \pi/2$ and $\phi_\theta = -\pi/2$, and take the difference:

$$\begin{aligned} I_{\pi/2} - I_{-\pi/2} &= |E_s|^2 + |E_R|^2 + 2E_s E_R \cos(\phi_s - \pi/2) && \text{(C.9)} \\ &\quad - \left(|E_s|^2 + |E_R|^2 + 2E_s E_R \cos(\phi_s + \pi/2)\right) && \text{(C.10)} \\ &= 4E_s E_R \sin(\phi_s) && \text{(C.11)} \\ &= 4E_R \mathrm{Im}(z_j). && \text{(C.12)} \end{aligned}$$

We find the real and imaginary components up to some arbitrary global scaling factor $4E_R$, which can be independently calculated by measuring the intensity of the reference region only. In experiment this is not necessary, due to the process of mapping optically-calculated values to theory using least-squares regression over many repeated random samples.

C.2 Real-valued measurements for multi-layer ONN

Here we detail the methods used to perform coherent detection of the real-valued MVM outputs for the two-layer ONN. We use two distinct methods: MVM-2a uses single-shot coherent detection similar to that already outlined in Sec. 3.1.2, whereas MVM-1 uses a new method of two sequential measurements.

We consider the forward propagating beam. Recall the cylindrical lens set CL-1 performs fan-in of MVM-1 by converging the beam in the horizontal direction, whilst

lens set CL-2b performs fan-in of MVM-2a by converging in the vertical direction. We create two dedicated regions on DMD-1 to produce two uniform reference beams, which pass through the system along with the forward ONN signal. The reference beam for the detection of the MVM-1 output is offset horizontally with respect to the signal beam, whereas that for the MVM-2a output is displaced vertically. In this way, the second reference beam does not interfere with the signal until the very last cylindrical lens set CL-2b, which acts to perform the MVM-2a summation as well as converge and mix the signal and second reference beam. For the second layer, the reference is ensured to be brighter than the signal, and because the two beams follow the same path they are phase-stable. This reference beam passes through both the slit and the activation vapor cell spatially separated from the signal, so experiences some attenuation but remains uniform, and does not interfere with the pump-probe process. As explained in Sec. 3.1.2, intensity measurement of the interfered beams at the ONN output gives $I_{\text{meas}} = |E_{\text{sig}} + E_{\text{ref}}|^2$ and the MVM-2a result can be read out in a single shot as $E_{\text{sig}} = \sqrt{I_{\text{meas}}} - E_{\text{ref}}$.

In contrast, due to the action of CL-1 the first reference beam does overlap and mix with the signal in the first slit, and they subsequently propagate together through the vapor cell. After the cell, the overlapping beams are tapped off via a beam splitter for measurement. Two measurements are performed sequentially, one to measure the amplitude, and a second to measure the sign. This is needed because the sum of signal and reference beams is operated on by the saturable absorption nonlinearity, and can no longer be linearly separated.

First the reference beam is turned off, and the signal intensity $I_{\text{meas1}} = |E_{\text{sig}}|^2$ without the reference beam is measured, which is used to yield the absolute value $|E_{\text{sig}}| = \sqrt{I_{\text{meas1}}}$. Second, the reference beam is turned on and the intensity $I_{\text{meas2}} = |E_{\text{sig}} + E_{\text{ref}}|^2$ is detected. In contrast to MVM-2a, the first reference beam is designed to be significantly weaker than the signal, and so comparing the magnitudes of I_{meas2} and I_{meas1} tells us whether E_{sig} is positive or negative.

However in practice this breaks down if the output vector takes values close to zero, smaller than the magnitude of the reference beam. In this case the sign

will be determined incorrectly, and indeed Fig. 4.11 in the main text shows some large errors around zero where the amplitude is measured correctly, but the sign is opposite to theory. However, in the optical training scheme, the nature of the SA nonlinearity is to suppress all values with small amplitude to zero. Therefore, by choosing a reference field amplitude smaller than the SA threshold value, these sign-flip errors are made redundant, as the sign is meaningless for zero amplitude, and Fig. 5.7 shows no such sign-flip errors occur.

D
Wavelength locking

The wavelength of our Toptica DL100 external cavity diode laser (ECDL) was locked to the chosen Rubidium atomic resonance by modulating the cavity length with a piezo-electric actuator. A feedback signal was generated and used to maintain the wavelength with a simple PID control circuit. The locking method used was derived from a dither-locking scheme [177].

Using a simple pump-probe spectroscopy method (see for example Fig. 4 in [178]), the atomic transitions of a secondary Rubidium vapor cell could be identified as a series of peaks similar to those shown in Fig. 5.4, by sending a *scanning* input signal to linearly displace the piezo. Next, a very high frequency, low amplitude *dither* signal was added to the piezo. Mixing the measured photocurrent response and original dither signal with analog electronics produced sum and difference frequencies, and applying a low pass filter to isolate the difference frequency yielded the *error* signal. This error signal was used to create a feedback signal using tunable analog PID electronics, which was applied to the piezo in order to maintain the laser frequency at the peak of one atomic transition. Full details of the method can be found in [177].

Hand-built electronics were used to generate the scanning and dither signals, amplify the photocurrent, perform signal mixing and filtering. The design schematic

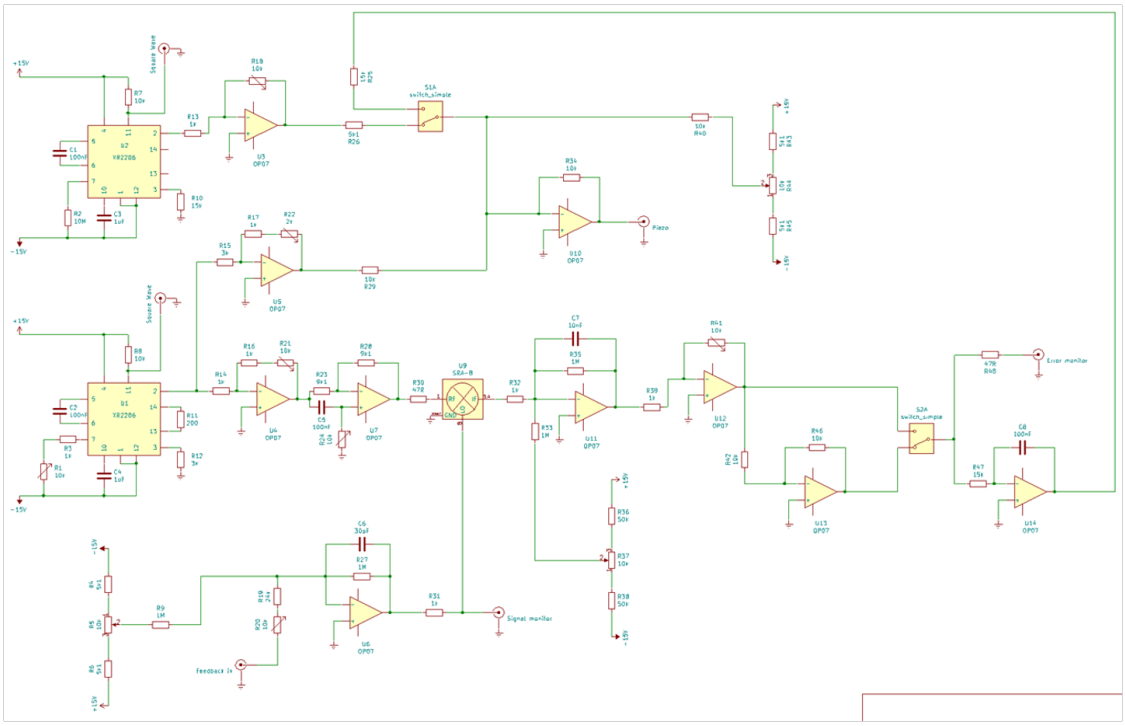


Figure D.1: Schematic design for frequency locking electronics, including generating scanning and dither signals, mixing and filtering, tuning gain and bias, switching signals, creating PID feedback, power delivery and signal amplification.

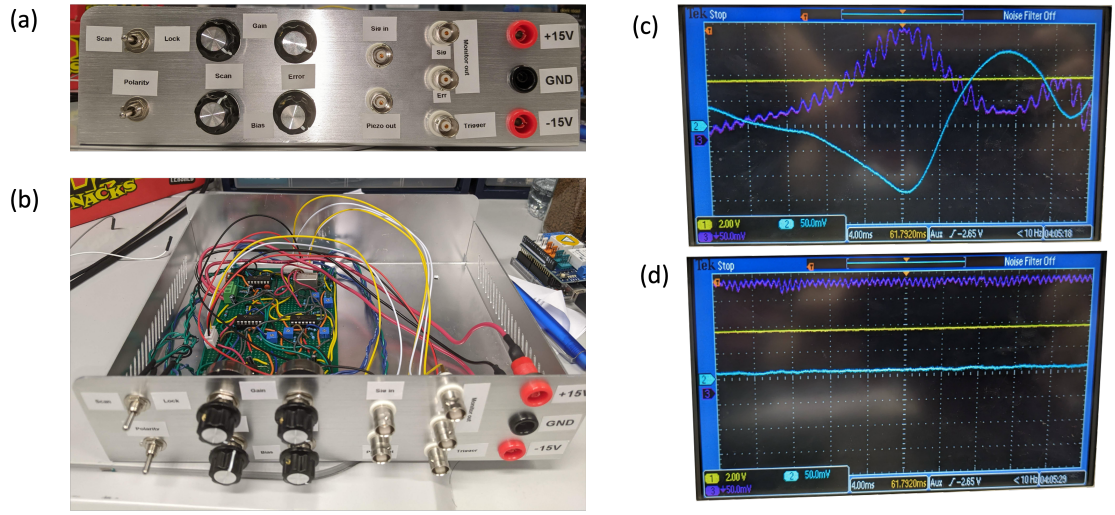


Figure D.2: Dither locking control and apparatus for laser frequency locking. (a) Front view of tuning controls, input and output connections and power supply. (b) Hand-made analog electronics. (c,d) Oscilloscope traces of photocurrent (purple) and error (blue) signals whilst applying the scanning and locking signals respectively to the piezo.

is shown in Fig. D.1. The electronics, tuning functions and signal input and outputs are shown in Fig. D.2(a,b).

The purple trace in Fig. D.2(c) shows the photocurrent from pump-probe spectroscopy as the piezo is scanned around the $F = 2 \rightarrow F' = 3$ transition peak. The dither signal has been amplified to show the effect. The blue trace shows the generated error signal. In Fig. D.2(d) the scanning signal is switched off, and the feedback signal applied to maintain the laser frequency at the transition peak.

Using this method, the laser wavelength was able to remain stable for up to an hour within the broadening width of the peak, estimated to be approximately 50 MHz, by comparison to the known theoretical frequency shift between the $F = 2 \rightarrow F' = 3$ and $F = 2 \rightarrow F' = 2$ transitions (267 MHz).

References

1. Jumper, J. *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* **596,** 583–589. doi:10.1038/s41586-021-03819-2 (2021).
2. Achiam, J. *et al.* GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774.* doi:https://doi.org/10.48550/arXiv.2303.08774 (2023).
3. Liu, Y. *et al.* Summary of ChatGPT-Related Research and Perspective Towards the Future of Large Language Models. *arXiv preprint arXiv:2304.01852.* doi:https://doi.org/10.48550/arXiv.2304.01852 (2023).
4. Liu, G. *et al.* Deep learning-guided discovery of an antibiotic targeting Acinetobacter baumannii. *Nature Chemical Biology.* doi:10.1038/s41589-023-01349-8 (2023).
5. Doerk, G. S. *et al.* Autonomous discovery of emergent morphologies in directed self-assembly of block copolymer blends. *Science Advances* **9.** doi:10.1126/sciadv.add3687 (2023).
6. Mankowitz, D. J. *et al.* Faster sorting algorithms discovered using deep reinforcement learning. *Nature* **618,** 257–263. doi:10.1038/s41586-023-06004-9 (2023).
7. Fawzi, A. *et al.* Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature* **610,** 47–53. doi:10.1038/s41586-022-05172-4 (2022).
8. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521,** 436–444. doi:10.1038/nature14539 (2015).
9. McCulloch, W. S. & Pitts, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics* **5,** 115–133. doi:10.1007/BF02478259 (1943).
10. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review* **65,** 386–408. doi:10.1037/h0042519 (1958).
11. Emmert-Streib, F. *et al.* An Introductory Review of Deep Learning for Prediction Models With Big Data. *Frontiers in Artificial Intelligence* **3,** 1–23. doi:10.3389/frai.2020.00004 (2020).
12. Hinton, G. E., Osindero, S. & Teh, Y.-W. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation* **18,** 1527–1554. doi:10.1162/neco.2006.18.7.1527 (2006).
13. Hecht-Nielsen. Theory of the backpropagation neural network. *International Joint Conference on Neural Networks* **1,** 593–605. doi:10.1109/IJCNN.1989.118638 (1989).

14. Sarker, I. H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science* **2,** 420. doi:10.1007/s42979-021-00815-1 (2021).

15. Sun, C. *et al.* Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. *Proceedings of the IEEE International Conference on Computer Vision,* 843–852. doi:10.1109/ICCV.2017.97 (2017).

16. Rawat, W. & Wang, Z. Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation* **29,** 2352–2449. doi:10.1162/NECO_a_00990 (2017).

17. Yu, Y. *et al.* A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation* **31,** 1235–1270. doi:10.1162/neco_a_01199 (2019).

18. Vaswani, A. *et al.* Attention Is All You Need. *arXiv preprint arXiv:1706.03762.* doi:https://doi.org/10.48550/arXiv.1706.03762 (2017).

19. Hoffmann, J. *et al.* Training Compute-Optimal Large Language Models. *arXiv preprint arXiv:2203.15556.* doi:https://doi.org/10.48550/arXiv.2203.15556 (2022).

20. Padua, D. *et al.* in *Encyclopedia of Parallel Computing* 1177–1184 (Springer US, Boston, 2011). doi:10.1007/978-0-387-09766-4_81.

21. Ball, P. The speed of computers. *Nature* **402,** 61. doi:10.1038/35011553 (1999).

22. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Communications of the ACM* **60,** 84–90. doi:10.1145/3065386 (2017).

23. Mittal, S. & Vaishay, S. A survey of techniques for optimizing deep learning on GPUs. *Journal of Systems Architecture* **99,** 101635. doi:10.1016/j.sysarc.2019.101635 (2019).

24. Shawahna, A., Sait, S. M. & El-Maleh, A. FPGA-Based accelerators of deep learning networks for learning and classification: A review. *IEEE Access* **7,** 7823–7859. doi:10.1109/ACCESS.2018.2890150 (2019).

25. Machupalli, R., Hossain, M. & Mandal, M. Review of ASIC accelerators for deep neural network. *Microprocessors and Microsystems* **89,** 104441. doi:10.1016/j.micpro.2022.104441 (2022).

26. Jouppi, N. P. *et al.* In-datacenter performance analysis of a tensor processing unit. *Proceedings - International Symposium on Computer Architecture,* 1–12. doi:10.1145/3079856.3080246 (2017).

27. Lohn, A. & Musser, M. AI and Compute: How Much Longer Can Computing Power Drive Artificial Intelligence Progress? *Center for Security and Emerging Technology.* doi:10.51593/2021CA009 (2022).

28. Sevilla, J. *et al.* Compute Trends Across Three Eras of Machine Learning. *Proceedings of the International Joint Conference on Neural Networks.* doi:10.1109/IJCNN55064.2022.9891914 (2022).

29. Waldrop, M. M. The chips are down for Moore's law. *Nature* **530,** 144–147. doi:10.1038/530144a (2016).

30. Powell, J. R. The quantum limit to Moore's law. *Proceedings of the IEEE* **96,** 1247–1248. doi:10.1109/JPROC.2008.925411 (2008).

31. Owens, J. D. *et al.* Research challenges for on-chip interconnection networks. *IEEE Micro* **27,** 96–108. doi:10.1109/MM.2007.4378787 (2007).

32. Amdahl, G. Validity of the single processor approach to achieving large scale computing capabilities. *AFIPS Spring joint computer conference,* 483–485. doi:https://doi.org/10.1145/1465482.1465560 (1967).

33. Heath, M. T. A tale of two laws. *The International Journal of High Performance Computing Applications* **29,** 320–330. doi:10.1177/1094342015572031 (2015).

34. Liu, Y. *et al.* Energy consumption and emission mitigation prediction based on data center traffic and PUE for global data centers. *Global Energy Interconnection* **3,** 272–282. doi:10.1016/j.gloei.2020.07.008 (2020).

35. Luccioni, A. S., Viguier, S. & Ligozat, A.-L. Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model. *arXiv preprint arXiv:2211.02001.* doi:https://doi.org/10.48550/arXiv.2211.02001 (2022).

36. Patterson, D. *et al.* Carbon Emissions and Large Neural Network Training. *arXiv preprint arXiv:2104.10350.* doi:https://doi.org/10.48550/arXiv.2104.10350 (2021).

37. Wu, J. *et al.* Analog Optical Computing for Artificial Intelligence. *Engineering* **10,** 133–145. doi:10.1016/j.eng.2021.06.021 (2022).

38. Solli, D. R. & Jalali, B. Analog optical computing. *Nature Photonics* **9,** 704–706. doi:10.1038/nphoton.2015.208 (2015).

39. Caulfield, H. J. & Dolev, S. Why future supercomputing requires optics. *Nature Photonics* **4,** 261–263. doi:10.1038/nphoton.2010.94 (2010).

40. Brunner, D. & Psaltis, D. Competitive photonic neural networks. *Nature Photonics* **15,** 323–324. doi:10.1038/s41566-021-00803-0 (2021).

41. Cheng, Q. *et al.* Recent advances in optical technologies for data centers: a review. *Optica* **5,** 1354. doi:10.1364/OPTICA.5.001354 (2018).

42. Ambs, P. Optical Computing: A 60-Year Adventure. *Advances in Optical Technologies,* 1–15. doi:10.1155/2010/372652 (2010).

43. Wakelin, S. & Walker, A. C. Digital optical computing. *Physics Education* **29,** 155–159. doi:10.1088/0031-9120/29/3/008 (1994).

44. Ready, J. F. in *Industrial Applications of Lasers* 559–589 (Elsevier, 1997). doi:10.1016/B978-012583961-7/50026-0.

45. Tucker, R. S. The role of optics in computing. *Nature Photonics* **4,** 405. doi:10.1038/nphoton.2010.162 (2010).

46. Goodman, J. W., Dias, A. R. & Woody, L. M. Fully parallel, high-speed incoherent optical method for performing discrete Fourier transforms. *Optics Letters* **2,** 1. doi:10.1364/ol.2.000001 (1978).

47. Weaver, C. S. & Goodman, J. W. A Technique for Optically Convolving Two Functions. *Applied Optics* **5,** 1248. doi:10.1364/AO.5.001248 (1966).

48. Farhat, N. H. *et al.* Optical implementation of the Hopfield model. *Applied Optics* **24,** 1469. doi:10.1364/ao.24.001469 (1985).

49. Tamura, N. & Wyant, J. C. Two-Dimensional Matrix Multiplication using Coherent Optical Techniques. *Optical Engineering* **18,** 198 (1979).

50. Abu-Mostafa, Y. S. & Psaltis, D. Optical Neural Computers. *Scientific American* **256,** 88–95. doi:10.2307/24979343 (1987).

51. Wagner, K. & Psaltis, D. Multilayer optical learning networks. *Applied Optics* **26,** 5061. doi:10.1364/AO.26.005061 (1987).

52. Psaltis, D., Brady, D. & Wagner, K. Adaptive optical networks using photorefractive crystals. *Applied Optics* **27,** 1752. doi:10.1364/ao.27.001752 (1988).

53. Jutamulia, S. & Yu, F. T. Overview of hybrid optical neural networks. *Optics and Laser Technology* **28,** 59–72. doi:10.1016/0030-3992(95)00070-4 (1996).

54. Shen, Y. *et al.* Deep learning with coherent nanophotonic circuits. *Nature Photonics* **11,** 441–446. doi:10.1038/nphoton.2017.93 (2017).

55. Lin, X. *et al.* All-optical machine learning using diffractive deep neural networks. *Science* **361,** 1004–1008. doi:10.1126/science.aat8084 (2018).

56. Zuo, Y. *et al.* All-optical neural network with nonlinear activation functions. *Optica* **6,** 1132. doi:10.1364/OPTICA.6.001132 (2019).

57. Zhou, T. *et al.* Large-scale neuromorphic optoelectronic computing with a reconfigurable diffractive processing unit. *Nature Photonics* **15,** 367–373. doi:10.1038/s41566-021-00796-w (2021).

58. Li, J. *et al.* Spectrally encoded single-pixel machine vision using diffractive networks. *Science Advances* **7.** doi:10.1126/sciadv.abd7690 (2021).

59. Zhang, H. *et al.* An optical neural chip for implementing complex-valued neural network. *Nature Communications* **12.** doi:10.1038/s41467-020-20719-7 (2021).

60. Wang, T. *et al.* An optical neural network using less than 1 photon per multiplication. *Nature Communications* **13.** doi:10.1038/s41467-021-27774-8 (2022).

61. Bandyopadhyay, S. *et al.* Single chip photonic deep neural network with accelerated training. *arXiv preprint arXiv:2208.01623.* doi:https://doi.org/10.48550/arXiv.2208.01623 (2022).

62. Chang, J. *et al.* Hybrid optical-electronic convolutional neural networks with optimized diffractive optics for image classification. *Scientific Reports* **8.** doi:10.1038/s41598-018-30619-y (2018).

63. Miscuglio, M. *et al.* Massively parallel amplitude-only Fourier neural network. *Optica* **7,** 1812. doi:10.1364/optica.408659 (2020).

64. Xu, X. *et al.* 11 TOPS photonic convolutional accelerator for optical neural networks. *Nature* **589,** 44–51. doi:10.1038/s41586-020-03063-0 (2021).

65. Feldmann, J. *et al.* Parallel convolutional processing using an integrated photonic tensor core. *Nature* **589,** 52–58. doi:10.1038/s41586-020-03070-1 (2021).

66. Bueno, J. *et al.* Reinforcement learning in a large-scale photonic recurrent neural network. *Optica* **5,** 756. doi:10.1364/optica.5.000756 (2018).

67. Van der Sande, G., Brunner, D. & Soriano, M. C. Advances in photonic reservoir computing. *Nanophotonics* **6,** 561–576. doi:10.1515/nanoph-2016-0132 (2017).

68. Feldmann, J. *et al.* All-optical spiking neurosynaptic networks with self-learning capabilities. *Nature* **569,** 208–214. doi:10.1038/s41586-019-1157-8 (2019).

69. Larger, L. *et al.* Photonic information processing beyond Turing: an optoelectronic implementation of reservoir computing. *Optics Express* **20,** 3241. doi:10.1364/oe.20.003241 (2012).

70. Lohmann, A. W. & Paris, D. P. Binary Fraunhofer Holograms, Generated by Computer. *Applied Optics* **6,** 1739. doi:10.1364/AO.6.001739 (1967).

71. Lee, W.-H. III Computer-Generated Holograms: Techniques and Applications. *Progress in Optics,* 119–232. doi:10.1016/S0079-6638(08)70072-6 (1978).

72. Ambs, P. *et al.* Computerized design and generation of space-variant holographic filters 1: System design considerations and applications of space-variant filters to image processing. *Applied Optics* **27,** 4753. doi:10.1364/AO.27.004753 (1988).

73. Lechner, B. *et al.* Liquid crystal matrix displays. *Proceedings of the IEEE* **59,** 1566–1579. doi:10.1109/PROC.1971.8489 (1971).

74. Labrunie, G., Robert, J. & Borel, J. A 128 × 128 electro-optical interface for real time data processing. *Revue de Physique Appliquée* **10,** 143–146. doi:10.1051/rphysap:01975001003014300 (1975).

75. Fisher, A. D. & Lee, J. N. The Current Status Of Two-Dimensional Spatial Light Modulator Technology. *Proc. SPIE: Optical and Hybrid Computing* **634,** 352–371. doi:10.1117/12.964024 (1986).

76. Reck, M. *et al.* Experimental realization of any discrete unitary operator. *Physical Review Letters* **73,** 58–61. doi:10.1103/PhysRevLett.73.58 (1994).

77. Clements, W. R. *et al.* Optimal design for universal multiport interferometers. *Optica* **3,** 1460. doi:10.1364/optica.3.001460 (2016).

78. Miller, D. A. B. Setting up meshes of interferometers – reversed local light interference method. *Optics Express* **25,** 29233. doi:10.1364/oe.25.029233 (2017).

79. Alexoudi, T. *et al.* Optics in computing: From photonic network-on-chip to chip-to-chip interconnects and disintegrated architectures. *Journal of Lightwave Technology* **37,** 363–379. doi:10.1109/JLT.2018.2875995 (2019).

80. Anderson, M. G. *et al.* Optical Transformers. *arXiv preprint arXiv:2302.10360.* doi:https://doi.org/10.48550/arXiv.2302.10360 (2023).

81. Bandyopadhyay, S., Hamerly, R. & Englund, D. Hardware error correction for programmable photonics. *Optica* **8,** 1247. doi:10.1364/optica.424052 (2021).

82. Hamerly, R., Bandyopadhyay, S. & Englund, D. Asymptotically fault-tolerant programmable photonics. *Nature Communications* **13,** 6831. doi:10.1038/s41467-022-34308-3 (2022).

83. Cheng, Z. *et al.* In-plane optical absorption and free carrier absorption in graphene-on-silicon waveguides. *IEEE Journal of Selected Topics in Quantum Electronics* **20,** 43–48. doi:10.1109/JSTQE.2013.2263115 (2014).

84. Wang, T. *et al.* Image sensing with multilayer nonlinear optical neural networks. *Nature Photonics* **17,** 408–415. doi:10.1038/s41566-023-01170-8 (2023).

85. Jha, A., Huang, C. & Prucnal, P. R. Reconfigurable all-optical nonlinear activation functions for neuromorphic photonics. *Optics Letters* **45,** 4819. doi:10.1364/OL.398234 (2020).

86. Williamson, I. *et al.* Reprogrammable Electro-Optic Nonlinear Activation Functions for Optical Neural Networks. *IEEE Journal of Selected Topics in Quantum Electronics* **26,** 1–12. doi:10.1109/JSTQE.2019.2930455 (2020).

87. Pour Fard, M. M. *et al.* Experimental realization of arbitrary activation functions for optical neural networks. *Optics Express* **28,** 12138. doi:10.1364/oe.391473 (2020).

88. Wetzstein, G. *et al.* Inference in artificial intelligence with deep optics and photonics. *Nature* **588,** 39–47. doi:10.1038/s41586-020-2973-6 (2020).

89. Hornik, K., Stinchcombe, M. & White, H. Multilayer feedforward networks are universal approximators. *Neural Networks* **2,** 359–366. doi:10.1016/0893-6080(89)90020-8 (1989).

90. Xu, R. *et al.* A survey of approaches for implementing optical neural networks. *Optics & Laser Technology* **136,** 106787. doi:10.1016/j.optlastec.2020.106787 (2021).

91. Shastri, B. J. *et al.* Photonics for artificial intelligence and neuromorphic computing. *Nature Photonics* **15,** 102–114. doi:10.1038/s41566-020-00754-y (2021).

92. Stroev, N. & Berloff, N. G. Analog Photonics Computing for Information Processing, Inference, and Optimization. *Advanced Quantum Technologies* **6.** doi:10.1002/qute.202300055 (2023).

93. De Marinis, L. *et al.* Photonic Neural Networks: A Survey. *IEEE Access* **7,** 175827–175841. doi:10.1109/ACCESS.2019.2957245 (2019).

94. Limbacher, B. *et al.* Terahertz optical machine learning for object recognition. *APL Photonics* **5.** doi:10.1063/5.0029310 (2020).

95. Zuo, Y. *et al.* Scalability of All-Optical Neural Networks Based on Spatial Light Modulators. *Physical Review Applied* **15,** 1. doi:10.1103/PhysRevApplied.15.054034 (2021).

96. Spall, J. *et al.* Fully reconfigurable coherent optical vector–matrix multiplication. *Optics Letters* **45,** 5752. doi:10.1364/OL.401675 (2020).

97. Hughes, T. W. *et al.* Training of photonic neural networks through in situ backpropagation and gradient measurement. *Optica* **5,** 864. doi:10.1364/optica.5.000864 (2018).

98. Pai, S. *et al.* Parallel Programming of an Arbitrary Feedforward Photonic Network. *IEEE Journal of Selected Topics in Quantum Electronics* **26.** doi:10.1109/JSTQE.2020.2997849 (2020).

99. Miller, D. A. B. Perfect optics with imperfect components. *Optica* **2,** 747. doi:10.1364/optica.2.000747 (2015).

100. Mercante, A. J. *et al.* Thin film lithium niobate electro-optic modulator with terahertz operating bandwidth. *Optics Express* **26,** 14810. doi:10.1364/oe.26.014810 (2018).

101. Miller, D. A. Attojoule Optoelectronics for Low-Energy Information Processing and Communications. *Journal of Lightwave Technology* **35,** 346–396. doi:10.1109/JLT.2017.2647779 (2017).

102. Hamerly, R. *et al.* Large-Scale Optical Neural Networks Based on Photoelectric Multiplication. *Physical Review X* **9.** doi:10.1103/PhysRevX.9.021032 (2019).

103. He, M. *et al.* High-performance hybrid silicon and lithium niobate Mach–Zehnder modulators for 100 Gbit s -1 and beyond. *Nature Photonics* **13,** 359–364. doi:10.1038/s41566-019-0378-6 (2019).

104. Tait, A. N. *et al.* Neuromorphic photonic networks using silicon photonic weight banks. *Scientific Reports* **7,** 7430. doi:10.1038/s41598-017-07754-z (2017).

105. Miscuglio, M. & Sorger, V. J. Photonic tensor cores for machine learning. *Applied Physics Reviews* **7.** doi:10.1063/5.0001942 (2020).

106. Huang, C. *et al.* Demonstration of scalable microring weight bank control for large-scale photonic integrated circuits. *APL Photonics* **5.** doi:10.1063/1.5144121 (2020).

107. Xu, X. *et al.* Photonic Perceptron Based on a Kerr Microcomb for High-Speed, Scalable, Optical Neural Networks. *Laser and Photonics Reviews* **14,** 1–10. doi:10.1002/lpor.202000070 (2020).

108. Shi, B., Calabretta, N. & Stabile, R. Deep Neural Network through an InP SOA-Based Photonic Integrated Cross-Connect. *IEEE Journal of Selected Topics in Quantum Electronics* **26,** 1–11. doi:10.1109/JSTQE.2019.2945548 (2020).

109. Ashtiani, F., Geers, A. J. & Aflatouni, F. An on-chip photonic deep neural network for image classification. *Nature* **606,** 501–506. doi:10.1038/s41586-022-04714-0 (2022).

110. Guo-Jie Li & Wah. The Design of Optimal Systolic Arrays. *IEEE Transactions on Computers* **C-34,** 66–77. doi:10.1109/TC.1985.1676516 (1985).

111. Zhou, W. *et al.* Phase-change materials for energy-efficient photonic memory and computing. *MRS Bulletin* **47,** 502–510. doi:10.1557/s43577-022-00358-7 (2022).

112. Zhou, T. *et al.* In situ optical backpropagation training of diffractive optical neural networks. *Photonics Research* **8,** 940. doi:10.1364/prj.389553 (2020).

113. Mengu, D. *et al.* Analysis of Diffractive Optical Neural Networks and Their Integration with Electronic Neural Networks. *IEEE Journal of Selected Topics in Quantum Electronics* **26.** doi:10.1109/JSTQE.2019.2921376 (2020).

114. Mengu, D., Rivenson, Y. & Ozcan, A. Scale-, Shift-, and Rotation-Invariant Diffractive Optical Networks. *ACS Photonics* **8,** 324–334. doi:10.1021/acsphotonics.0c01583 (2021).

115. Yan, T. *et al.* Fourier-space Diffractive Deep Neural Network. *Physical Review Letters* **123,** 23901. doi:10.1103/PhysRevLett.123.023901 (2019).

116. Rahman, M. S. S. *et al.* Ensemble learning of diffractive optical networks. *Light: Science and Applications* **10.** doi:10.1038/s41377-020-00446-w (2021).

117. Li, J. *et al.* Class-specific differential detection in diffractive optical neural networks improves inference accuracy. *Advanced Photonics* **1,** 1–13. doi:10.1117/1.AP.1.4.046001 (2019).

118. Luo, Y. *et al.* Design of task-specific optical systems using broadband diffractive neural networks. *Light: Science and Applications* **8,** 1–14. doi:10.1038/s41377-019-0223-1 (2019).

119. Lu, L. *et al.* Miniaturized Diffraction Grating Design and Processing for Deep Neural Network. *IEEE Photonics Technology Letters* **31,** 1952–1955. doi:10.1109/LPT.2019.2948626 (2019).

120. Qian, C. *et al.* Dynamic recognition and mirage using neuro-metamaterials. *Nature Communications* **13,** 2694. doi:10.1038/s41467-022-30377-6 (2022).

121. McMahon, P. L. The physics of optical computing. *Nature Reviews Physics* **5,** 717–734. doi:10.1038/s42254-023-00645-5 (2023).

122. Goodman, J. W. Fan-in and Fan-out with Optical Interconnections. *Optica Acta: International Journal of Optics* **32,** 1489–1496. doi:10.1080/713821684 (1985).

123. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations,* 1–15 (2014).

124. Buckley, S. M. *et al.* Photonic online learning: a perspective. *Nanophotonics* **12,** 833–845. doi:10.1515/nanoph-2022-0553 (2023).

125. Wright, L. G. *et al.* Deep physical neural networks trained with backpropagation. *Nature* **601,** 549–555. doi:10.1038/s41586-021-04223-6 (2022).

126. Mouret, J. B. & Chatzilygeroudis, K. 20 Years of reality gap: A few thoughts about simulators in evolutionary robotics. *Proceedings of the Genetic and Evolutionary Computation Conference Companion,* 1121–1124. doi:10.1145/3067695.3082052 (2017).

127. Vadlamani, S. K., Englund, D. & Hamerly, R. Transferable learning on analog hardware. *Science Advances* **9.** doi:10.1126/sciadv.adh3436 (2023).

128. Gerchberg, R. W. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik* **35,** 237–246 (1972).

129. Andreoli, L. *et al.* Boolean learning under noise-perturbations in hardware neural networks. *Nanophotonics* **9,** 4139–4147. doi:10.1515/nanoph-2020-0171 (2020).

130. Schuman, C. D. *et al.* A Survey of Neuromorphic Computing and Neural Networks in Hardware. *arXiv preprint arXiv:1705.06963.* doi:https://doi.org/10.48550/arXiv.1705.06963 (2017).

131. Nøkland, A. Direct Feedback Alignment Provides Learning in Deep Neural Networks. *Advances in Neural Information Processing Systems,* 1045–1053. doi:https://doi.org/10.48550/arXiv.1609.01596 (2016).

132. Filipovich, M. J. *et al.* Silicon photonic architecture for training deep neural networks with direct feedback alignment. *Optica* **9,** 1323. doi:10.1364/OPTICA.475493 (2022).

133. Buckley, S. & McCaughan, A. A general approach to fast online training of modern datasets on real neuromorphic systems without backpropagation. *Proceedings of the International Conference on Neuromorphic Systems,* 1–8. doi:10.1145/3546790.3546810 (2022).

134. Bartunov, S. *et al.* Assessing the Scalability of Biologically-Motivated Deep Learning Algorithms and Architectures. *Advances in Neural Information Processing Systems,* 9368–9378. doi:https://doi.org/10.48550/arXiv.1807.04587 (2018).

135. Bengio, Y. *Practical Recommendations for Gradient-Based Training of Deep Architectures* 437–478. doi:10.1007/978-3-642-35289-8_26 (Springer Berlin Heidelberg, 2012).

136. LeCun, Y. A. *et al.* in *Neural Networks: Tricks of the Trade* 9–48 (Springer Berlin Heidelberg, 2012). doi:10.1007/978-3-642-35289-8_3.

137. Spall, J., Guo, X. & Lvovsky, A. I. Hybrid training of optical neural networks. *Optica* **9,** 803. doi:10.1364/OPTICA.456108 (2022).

138. Tam, S. *et al.* Learning on an analog VLSI neural network chip. *IEEE International Conference on Systems, Man, and Cybernetics,* 701–703. doi:10.1109/ICSMC.1990.142209 (1990).

139. Mennel, L. *et al.* Ultrafast machine vision with 2D material neural network image sensors. *Nature* **579,** 62–66. doi:10.1038/s41586-020-2038-x (2020).

140. Li, C. *et al.* Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. *Nature Communications* **9,** 2385. doi:10.1038/s41467-018-04484-2 (2018).

141. Wang, Z. *et al.* In situ training of feed-forward and recurrent convolutional memristor networks. *Nature Machine Intelligence* **1,** 434–442. doi:10.1038/s42256-019-0089-1 (2019).

142. Pai, S. *et al.* Experimentally realized in situ backpropagation for deep learning in photonic neural networks. *Science* **380,** 398–404. doi:10.1126/science.ade8450 (2023).

143. Guo, X. *et al.* Backpropagation through nonlinear units for the all-optical training of neural networks. *Photonics Research* **9,** 71. doi:10.1364/prj.411104 (2021).

144. Monk, D. W. in *Handbook of Visual Display Technology* 2081–2093 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012). doi:10.1007/978-3-540-79567-4_129.

145. Schonbrun, E. *et al.* 3D interferometric optical tweezers using a single spatial light modulator. *Optics Express* **13,** 3777. doi:10.1364/opex.13.003777 (2005).

146. Hu, L. *et al.* Phase-only liquid crystal spatial light modulator for wavefront correction with high precision. *Optics Express* **12,** 6403. doi:10.1364/opex.12.006403 (2004).

147. Ahderom, S. *et al.* Applications of liquid crystal spatial light modulators in optical communications. *5th IEEE International Conference on High Speed Networks and Multimedia Communications,* 239–242. doi:10.1109/HSNMC.2002.1032583 (2002).

148. Chen, H. M. P. *et al.* Pursuing high quality phase-only liquid crystal on silicon (LCoS) devices. *Applied Science* **8,** 2323. doi:10.3390/app8112323 (2018).

149. Xun, X. & Cohn, R. W. Phase calibration of spatially nonuniform spatial light modulators. *Applied Optics* **43,** 6400–6406. doi:10.1364/AO.43.006400 (2004).

150. Moser, S., Ritsch-Marte, M. & Thalhammer, G. Model-based compensation of pixel crosstalk in liquid crystal spatial light modulators. *Optics Express* **27,** 25046. doi:10.1364/oe.27.025046 (2019).

151. Pushkina, A. A. *et al.* Comprehensive model and performance optimization of phase-only spatial light modulators. *Measurement Science and Technology* **31,** 125202. doi:10.1088/1361-6501/aba56b (2020).

152. Engström, D. *et al.* Calibration of spatial light modulators suffering from spatially varying phase response. *Optics Express* **21,** 16086. doi:10.1364/oe.21.016086 (2013).

153. Arrizón, V., Méndez, G. & Sánchez-de-La-Llave, D. Accurate encoding of arbitrary complex fields with amplitude-only liquid crystal spatial light modulators. *Optics Express* **13,** 7913. doi:10.1364/opex.13.007913 (2005).

154. Arrizón, V. *et al.* Pixelated phase computer holograms for the accurate encoding of scalar complex fields. *Journal of the Optical Society of America A* **24,** 3500–3507. doi:10.1364/josaa.24.003500 (2007).

155. Mendoza-Yero, O., Mínguez-Vega, G. & Lancis, J. Encoding complex fields by using a phase-only optical element. *Optics Letters* **39,** 1740. doi:10.1364/ol.39.001740 (2014).

156. Bolduc, E. *et al.* Exact solution to simultaneous intensity and phase encryption with a single phase-only hologram. *Optics Letters* **38,** 3546. doi:10.1364/ol.38.003546 (2013).

157. Clark, T. W. *et al.* Comparison of beam generation techniques using a phase only spatial light modulator. *Optics Express* **24,** 6249. doi:10.1364/oe.24.006249 (2016).

158. Li Deng. The MNIST Database of Handwritten Digit Images for Machine Learning Research. *IEEE Signal Processing Magazine* **29,** 141–142. doi:10.1109/MSP.2012.2211477 (2012).

159. Starasotnikau, M. A. Assessment of Temperature Effects in Interior Orientation Parameters Calibration of Optoelectronic Devices. *Devices and Methods of Measurements* **11,** 122–131. doi:10.21122/2220-9506-2020-11-2-122-131 (2020).

160. Spall, J., Guo, X. & Lvovsky, A. I. Training neural networks with end-to-end optical backpropagation. *arXiv preprint arXiv:2308.05226.* doi:https://doi.org/10.48550/arXiv.2308.05226 (2023).

161. Preston, D. W. Doppler-free saturated absorption: Laser spectroscopy. *American Journal of Physics* **64,** 1432–1436 (1996).

162. Olivares, I. E. & Carrazana, P. in *Techniques for Lithium Isotope Separation, Laser Cooling, and Scattering* 2–1 to 2–17 (IOP Publishing, 2022). doi:10.1088/978-0-7503-3839-4ch2.

163. Wang, G., Baker-Murray, A. A. & Blau, W. J. Saturable Absorption in 2D Nanomaterials and Related Photonic Devices. *Laser & Photonics Reviews* **13,** 1–23. doi:10.1002/lpor.201800282 (2019).

164. Hooker, S. & Webb, C. E. *Laser physics* (Oxford University Press, 2010).

165. Steck, D. A. Rubidium 87 D Line Data. *http://steck.us/alkalidata (revision 2.3.2, 10 September 2023).* doi:http://steck.us/alkalidata (2023).

166. Chen, X. *et al.* Symbolic Discovery of Optimization Algorithms. *arXiv preprint arXiv:2302.06675.* doi:https://doi.org/10.48550/arXiv.2302.06675 (2023).

167. Boyd, R. W. *Nonlinear Optics* doi:10.1016/C2015-0-05510-1 (Elsevier, 2020).

168. Gholami, A. *et al.* in *Low-Power Computer Vision* 291–326 (Chapman and Hall/CRC, Boca Raton, 2022). doi:10.1201/9781003162810-13.

169. Wang, M. *et al.* NITI: Training Integer Neural Networks Using Integer-Only Arithmetic. *IEEE Transactions on Parallel and Distributed Systems* **33,** 3249–3261. doi:10.1109/TPDS.2022.3149787 (2022).

170. Hubara, I. *et al.* Quantized neural networks: Training neural networks with low precision weights and activations. *Journal of Machine Learning Research* **18,** 1–30. doi:https://doi.org/10.48550/arXiv.1609.07061 (2018).

171. Zhang, W. *et al.* Silicon microring synapses enable photonic deep learning beyond 9-bit precision. *Optica* **9,** 579. doi:10.1364/OPTICA.446100 (2022).

172. Ma, S.-Y. *et al.* Quantum-noise-limited optical neural networks operating at a few quanta per activation (2023).

173. Panuski, C. L. *et al.* A full degree-of-freedom spatiotemporal light modulator. *Nature Photonics* **16,** 834–842. doi:10.1038/s41566-022-01086-9 (2022).

174. Feng, N.-N. *et al.* High speed carrier-depletion modulators with 14V-cm V_πL integrated on 025μm silicon-on-insulator waveguides. *Optics Express* **18,** 7994. doi:10.1364/oe.18.007994 (2010).

175. Xu, N. *et al.* Recent advances in nano-opto-electro-mechanical systems. *Nanophotonics* **10,** 2265–2281. doi:10.1515/nanoph-2021-0082 (2021).

176. James, G. *et al. An Introduction to Statistical Learning* doi:10.1007/978-1-4614-7138-7 (Springer New York, New York, NY, 2013).

177. Juarez, M. The Use of a Lock-In Amplifier to Stabilize the Frequency of a Diode Laser. *College Honors Program* **9.** doi:https://crossworks.holycross.edu/honors/9 (2009).

178. Himsworth, M. & Freegarde, T. Rubidium pump-probe spectroscopy: Comparison between ab initio theory and experiment. *Physical Review A - Atomic, Molecular, and Optical Physics* **81,** 23423. doi:10.1103/PhysRevA.81.023423 (2010).